Sécurité informatique

Principes et méthode

Laurent Bloch

Christophe Wolfhugel

Préfaces de Christian Queinnec et Hervé Schauer

Avec la contribution de Nat Makarévitch



ÉDITIONS EYROLLES
61, bld Saint-Germain
75240 Paris Cedex 05
www.editions-eyrolles.com

Chez le même éditeur

T. Limoncelli, adapté par S. Blondeel. – **Admin'sys. Gérer son temps…** N°11957, 2006, 274 pages.

C. Llorens, L. Levier, D. Valois. – **Tableaux de bord de la sécurité réseau.** N°11973, 2006, 560 pages.

J. Steinberg, T. Speed, adapté par B. Sonntag. – **SSL VPN.** *Accès web et extranets sécurisés.* N°11933, 2006, 220 pages.

I. Hurbain, avec la contribution d'E. Dreyfus. – **Mémento Unix/Linux.** N°11954, 2006, 14 pages.

M. Bäck *et al.*, adapté par P. Tonnerre – **Monter son serveur de mails sous Linux.** N°11931, 2006, 360 pages.

M. Krafft, adapté par R. Hertzog, R. Mas, dir. N. Makarévitch. – **Debian.** *Administration et configuration avancées.* N°11904, 2006, 674 pages.

R. Hertzog, C. Le Bars, R. Mas. – **Cahier de l'admin Debian, 2e édition.** N°11639, 2005, 310 pages.

B. Boutherin, B. Delaunay. – **Sécuriser un réseau Linux, 3e édition.** N°11960, 2007, 250 pages.

C. Blaess. – **Programmation système en C sous Linux.** N°11601, 2e édition 2005, 964 pages.

J Battelle, trad. D. Rueff, avec la contribution de S. Blondeel – **La révolution Google.** N°11903, 2006, 280 pages.

L. Dricot, avec la contribution de R. Mas. – **Ubuntu efficace.** N°12003, 2e édition 2006, 360 pages avec CD-Rom.

Préfaces de Christian Queinnec et d'Hervé Schauer.

Avec la contribution de Solveig, Florence Henry et Nat Makarévitch.



Préface I

L'internet, on le lit souvent, est une jungle, un lieu plein de dangers sournois, tapis et prêts à frapper fort, péniblement et durablement. On aura intérêt à ne pas s'attarder sur cette banalité car la jungle est l'endroit où l'on doit obligatoirement vivre. En revanche, tout comme pour les MST (maladies sexuellement transmissibles), être ignare, ne pas vouloir apprécier les dangers, persister à les ignorer sont des attitudes blâmables.

Ce qui est moins évident est qu'un ordinateur est un assemblage hétéroclite, historiquement enchevêtré, de matériels et de logiciels dont les innombrables interactions sont au-delà de l'humainement appréhendable. Un ordinateur est tout autant une jungle et cette jungle s'étend au fil de ses expositions successives à Internet.

Comme dans tant d'autres domaines sociétaux, la « sécurité » est réputée être la solution à ces problèmes !

Mais au-delà de bonnement nommer cette solution, d'espérer un monde meilleur où l'on pourrait enfin jouir de cette fameuse sécurité, il faut s'interroger sur son existence, sa nature, ses constituants, ses conditions d'apparition ou de disparition, son développement, etc. C'est précisément à ces interrogations que répond l'ouvrage de Laurent Bloch et Christophe Wolfhugel.

Un tel état de grâce ne s'obtient ni par décret, ni par hasard. C'est le résultat d'une confluence opiniâtre de comportements et de solutions techniques. Ces dernières sont présentes depuis l'article séminal de Diffie et Hellman [40] en 1976 qui a permis de résoudre le problème, ouvert depuis des millénaires : comment

deux personnes, ne se connaissant au préalable pas, peuvent-elles élaborer un secret commun à elles seules en n'ayant échangé que des messages publics ? Clés publiques/privées, certificat, signature électronique sont les plus connues des innovations qui en découlent. Notariat électronique, respect de l'anonymat (réseau TOR), crypto-virus blindés en sont d'autres plus confidentielles aujourd'hui.

Si, dès maintenant, des solutions techniques existent pour un monde meilleur, c'est sur le plan humain que peine le salut à s'instaurer. On ne peut souhaiter un monde sûr que si l'on prend la mesure de l'actuelle insécurité. On ne peut espérer un monde plus sûr que si l'on sait qu'il est réalisable, que si l'on est prêt à tolérer des changements personnels importants de comportement, que si l'entière société humaine stimule l'adoption de ces nouvelles règles et veille à les adapter au rythme des inéluctables évolutions.

La sécurité n'est qu'un sentiment dont l'éclosion est due à la conjonction de facteurs techniques et sociétaux. La mise en place d'un contexte favorable à ce sentiment est complexe, tant sont grandes les difficultés de réalisation et les oppositions entre les différentes inclinations libertaires, dirigistes ou *Big Brother*-iennes. L'anonymat est-il autorisé sur Internet ? Puis-je mettre mon ordinateur en conformité avec mes désirs sécuritaires ? Comment rétribuer une création intellectuelle incarnée numériquement (*sic*) et dont la duplication est quasiment gratuite ? Devons-nous laisser l'offre des industriels diriger notre morale et notre liberté ?

L'excellent livre de Laurent Bloch et Christophe Wolfhugel a pour thème la sécurité. Loin de s'appesantir sur les seuls aspects techniques ou de broder autour de banalités comme « la sécurité parfaite n'existe pas » ou encore « avoir listé les menaces garantit une éternelle quiétude », ce livre est à lire et à méditer pour tous ceux qui y croient et tous ceux qui n'y croient pas afin que tous puissent participer intelligemment à l'avènement de l'ère numérique. Cet espace incommensurablement démocratique (les internautes votent avec leur souris) que réalise l'interconnexion de toutes les puces calculantes, nous avons une chance de modeler son avenir tout autant que de le transformer en le plus effroyablement fliqué lieu communautaire. À nous de choisir à la lueur de ce que nous en dit cet ouvrage.

Christian Queinnec
Professeur à l'Université
Pierre et Marie Curie

Préface II

Alors que la sécurité des systèmes d'information était un produit de luxe, elle tend aujourd'hui à devenir un moyen d'apporter la confiance au cœur des affaires.

Cet ouvrage en rappelle les bases techniques et présente une perspective nouvelle, pertinente et utile à tous les acteurs du secteur de la sécurité des systèmes d'information, par deux esprits vifs, qui ont prouvé, par leur carrière et leurs réalisations, leur indépendance et leur compétence.

Hervé Schauer
Consultant en sécurité
des systèmes d'informations
depuis 1989

Table des matières

Avant-propos

Ce livre procurera au lecteur les connaissances de base en sécurité informatique dont aucun utilisateur d'ordinateur ni aucun internaute ne devrait être dépourvu, qu'il agisse dans le cadre professionnel ou à titre privé. Pour cela nous lui proposerons quelques pistes qui devraient l'aider à trouver son chemin dans un domaine en évolution rapide où l'information de qualité est parfois difficile à distinguer du vacarme médiatique et des rumeurs sans fondement.

Plutôt que de proposer des recettes à appliquer telles quelles, et qui dans un domaine en évolution rapide seraient de toute façon vouées à une prompte péremption, nous présenterons des axes de réflexion accompagnés d'exemples techniques.

L'Internet est au cœur des questions de sécurité informatique : nous rappellerons brièvement ses principes de fonctionnement, placés sous un éclairage qui fera apparaître les risques qui en découlent. Pas de sûreté de fonctionnement sans un bon système d'exploitation : nous passerons en revue les qualités que nous sommes en droit d'en attendre. Nous examinerons les différentes formes de malfaisance informatique, sans oublier les aspects organisationnels et sociaux de la sécurité. Pour les entreprises, nous proposerons quelques modèles de documents utiles à l'encadrement des activités informatiques de leur personnel.

La protection des systèmes d'information repose aujourd'hui sur la cryptographie : nous donnerons un exposé aussi simple que possible des principes de cette science, qui permette au lecteur qui le souhaite d'en comprendre les bases mathématiques, cependant que celui qui serait rebuté par ces aspects pourra en première lecture sauter sans (trop) de dommage ces développements.

Nous terminerons par un tour d'horizon des nouvelles possibilités de l'Internet, qui engendrent autant de nouveaux risques : échange de fichiers *peer to peer*, téléphonie sur IP avec des systèmes tels que *Skype*.

Les lignes qui suivent sont avant tout le fruit de nos expériences professionnelles respectives, notamment dans les fonctions de responsable de la sécurité des systèmes d'information de l'Institut National de la Santé et de la Recherche Médicale (INSERM) pour l'un, d'expert des protocoles de l'Internet au sein de la division Orange Business Services de France Télécom pour l'autre.

L'informatique en général, ses domaines techniques plus que les autres, et celui de la sécurité tout particulièrement, sont envahis de « solutions », que des entreprises s'efforcent de vendre à des clients qui pourraient être tentés de les acheter avant d'avoir identifié les problèmes qu'elles sont censées résoudre. Il est vrai que la démarche inductive est souvent fructueuse dans les domaines techniques, et que la démonstration d'une solution ingénieuse peut faire prendre conscience d'un problème, et du coup aider à sa solution. Mais l'induction ne peut trouver son chemin que dans un esprit déjà fécondé par quelques interrogations : le but des lignes qui suivent est de contribuer à cet effort de réflexion.

L'axe de ce livre, on l'aura compris, n'est pas dirigé vers les modes d'emploi de logiciels ou de matériels de sécurité, mais bien plutôt vers la position et l'explication des problèmes de sécurité, insérés dans un contexte technique dont il faut comprendre les tenants et les aboutissants si l'on veut adopter des solutions raisonnables. Et donner dans un livre des solutions techniques ou, pire, des recettes toutes faites, nous semblerait futile à une heure où le contexte technique évolue si vite que le Web et la presse spécialisée (qui se développe, y compris en langue française, cf. par exemple la revue MISC [4]) nous semblent bien mieux placés pour répondre à ce type d'attente. Il nous a paru plus judicieux de proposer au lecteur un tour d'horizon des problèmes afin qu'il puisse plus facilement, le moment venu, choisir entre plusieurs solutions techniques qui pourraient s'offrir à lui face à un problème concret.

Mode d'emploi du livre

Comment aborder la lecture de ce livre ? Il propose une progression des explications. Pour le chiffrement, qui est le point le plus difficile parce qu'assez technique mais à la base de tout le reste, il y a d'abord une évocation informelle et succincte

(chapitre 1), puis une présentation générale de la fonction de chiffrement, sans préjuger de ce qu'elle est (chapitre 2), puis l'explication précise avec exposé mathématique (chapitre 4). Il semble difficile de faire autrement, parce que certains lecteurs ont le droit de ne pas lire les mathématiques du chapitre 4, mais ils ont le droit de comprendre le reste quand même. Mettre l'explication complète au début risquerait de décourager le lecteur, supprimer l'explication préalable du chapitre 2 est logiquement impossible parce que ce serait saper les développements qui suivent. Le prix de cette progression est qu'il y a des *flashbacks* : nous pensons qu'il vaut mieux revenir sur un sujet que d'égarer le lecteur par une attaque trop abrupte.

Conventions typographiques

Les textes encadrés ainsi sont destinés à des explications plus techniques que les autres passages, à des exemples pratiques ou à des apartés.

Les nombres entre crochets comme ceci [24] renvoient aux entrées de la bibliographie, en fin de volume.

Le livre comporte quatre parties, qui nous semblent correspondre aux quatre axes selon lesquels un responsable de sécurité doit déployer ses compétences et son activité :

- la première partie expose les principes généraux de sécurité, de façon aussi peu technique que possible ; vous devriez pouvoir la faire lire à votre directeur du système d'information ;
- la seconde partie, consacrée à la *science de la sécurité du système d'information*, présente les bases scientifiques sur lesquelles reposent les techniques pratiques ; elle est plus exigeante pour le lecteur en termes de difficulté conceptuelle ;
- la troisième partie aborde les aspects politiques, sociaux et psychologiques de la sécurité ; vous devriez pouvoir la placer sous les yeux de votre directeur juridique et de votre DRH ;
- la quatrième partie, qui envisage les évolutions récentes des menaces et de la sécurité, devrait intéresser quiconque navigue régulièrement sur l'Internet.

Remerciements

La liste de tous ceux à qui ce livre doit quelque chose serait trop longue pour que nous prenions le risque, en la dressant, d'en oublier trop. Nous citerons Dominique Sabrier, pour ses relectures toujours précises et d'une exigence judicieuse. L'idée de ce livre naquit d'un enseignement de master organisé à l'université Paris 12 par Alexis Bes. Christian Queinnec (outre sa préface), Michel Gaudet, Bernard Perrot, Patrick Lerouge, Nat Makarévitch et Solveig ont relu, utilement commenté, conseillé et encouragé. Nos collègues de l'INSERM et de France Télécom, sans en avoir forcément eu conscience, ont aussi contribué tant par les échanges d'expériences et d'avis que par les situations concrètes soumises à notre examen. Muriel Shan Sei Fan fut une éditrice à l'exigence stimulante. Florence Henry a mis à la composition la touche finale qui fait l'esthétique de l'ouvrage. Les activités et réunions organisées par l'Observatoire de la sécurité des systèmes d'information et des réseaux (OSSIR), par le Symposium sur la sécurité des technologies de l'information et de la communication (SSTIC) et par les Journées réseau de l'enseignement supérieur (JRES) furent des sources d'inspiration permanentes : parmi les intervenants, nous citerons notamment Éric Filiol, Nicolas Ruff, Hervé Schauer. Je remercie François Bayen pour ses suggestions qui ont amélioré notablement les exposés cryptographiques du chapitre 4. La responsabilité des erreurs qui subsistent néanmoins dans ce texte ne peut être imputée qu'aux auteurs.

Ce livre a été écrit, composé et mis en page au moyen de logiciels libres, notamment Linux, GNU/Emacs, TeX, LaTeX, BibTeX et `xfig` : il convient d'en remercier ici les auteurs et contributeurs, dont le travail désintéressé élargit le champ de la liberté d'expression.

Première partie

Principes de sécurité du système d’information

1

Premières notions de sécurité

Ce chapitre introduit les notions de base de la sécurité informatique : menace, risque, vulnérabilité ; il effectue un premier parcours de l'ensemble du domaine, de ses aspects humains, techniques et organisationnels, sans en donner de description technique.

Menaces, risques, vulnérabilités

La Sécurité des Systèmes d'Information (SSI) est aujourd'hui un sujet important parce que le système d'information (SI) est pour beaucoup d'entreprises un élément absolument vital : le lecteur de ce livre, *a priori*, devrait être déjà convaincu de de cette évidence, mais il n'est peut-être pas inutile de lui donner quelques munitions pour l'aider à en convaincre sa hiérarchie. Il pourra par exemple à cet effet consulter le livre de Michel Volle *e-conomie* [114], disponible en ligne, qui explique comment pour une entreprise comme *Air-France* le SI, qui comporte notamment

le système de réservation *Amadeus*, est un actif plus crucial que les avions. En effet, toutes les compagnies font voler des avions : mais la différence entre celles qui survivent et celles qui disparaissent (rappelons l'hécatombe récente : Panam, TWA, Swissair, Sabena...) réside d'une part dans l'aptitude à optimiser l'emploi du temps des avions et des équipages, notamment par l'organisation de *hubs*, c'est-à-dire de plates-formes où convergent des vols qui amènent des passagers qui repartiront par d'autres vols de la compagnie, d'autre part dans l'aptitude à remplir les avions de passagers qui auront payé leur billet le plus cher possible, grâce à la technique du *yield management*, qui consiste à calculer pour chaque candidat au voyage le prix à partir duquel il renoncerait à prendre l'avion, et à lui faire payer juste un peu moins. Ce qui permet aux compagnies d'atteindre ces objectifs, et ainsi de l'emporter sur leurs rivales, c'est bien leur SI, qui devient dès lors un outil précieux, irremplaçable, en un mot vital.

La même chose est déjà vraie depuis longtemps pour les banques, bien sûr.

Puisque le SI est vital, tout ce qui le menace est potentiellement mortel. Conjurer les menaces contre le SI est devenu impératif, et les lignes qui suivent sont une brève description de ce qu'il faut faire pour cela.

Les menaces contre le système d'information entrent dans une des catégories suivantes : atteinte à la disponibilité des systèmes et des données, destruction de données, corruption ou falsification de données, vol ou espionnage de données, usage illicite d'un système ou d'un réseau, usage d'un système compromis pour attaquer d'autres cibles.

Les menaces engendrent des risques et coûts humains et financiers : perte de confidentialité de données sensibles, indisponibilité des infrastructures et des données, dommages pour le patrimoine intellectuel et la notoriété. Les risques peuvent se réaliser si les systèmes menacés présentent des vulnérabilités.

Il est possible de préciser la notion de risque en la décrivant comme le produit d'un préjudice par une probabilité d'occurrence :

```
risque = préjudice x probabilité d'occurrence
```

Cette formule exprime qu'un événement dont la probabilité est assez élevée, par exemple la défaillance d'un disque dur, mais dont il est possible de prévenir le préjudice qu'il peut causer, par des sauvegardes régulières, représente un risque acceptable ; il en va de même pour un événement à la gravité imparable, comme

l'impact d'un météorite de grande taille, mais à la probabilité d'occurrence faible. Il va de soi que, dans le premier cas, le risque ne devient acceptable que si les mesures de prévention contre le préjudice sont effectives et efficaces : cela irait sans dire, si l'oubli de cette condition n'était très fréquent (cf. page 17).

Si la question de la sécurité des systèmes d'information a été radicalement bouleversée par l'évolution rapide de l'Internet, elle ne saurait s'y réduire ; il s'agit d'un vaste problème dont les aspects techniques ne sont qu'une partie. Les aspects juridiques, sociaux, ergonomiques, psychologiques et organisationnels sont aussi importants, sans oublier les aspects immobiliers, mais nous commencerons par les aspects techniques liés à l'informatique.

Aspects techniques de la sécurité

Les problèmes techniques actuels de sécurité informatique peuvent, au moins provisoirement, être classés en deux grandes catégories :

- ceux qui concernent la sécurité de l'ordinateur proprement dit, serveur ou poste de travail, de son système d'exploitation et des données qu'il abrite ;
- ceux qui découlent directement ou indirectement de l'essor des réseaux, qui multiplie la quantité et la gravité des menaces.

Si les problèmes de la première catégorie citée ici existent depuis la naissance de l'informatique, il est clair que l'essor des réseaux, puis de l'Internet, en a démultiplié l'impact potentiel en permettant leur combinaison avec ceux de la seconde catégorie.

La résorption des vulnérabilités repose sur un certain nombre de principes et de méthodes que nous allons énumérer dans la présente section avant de les décrire plus en détail.

Définir risques et objets à protéger

Périmètre de sécurité

Inutile de se préoccuper de sécurité sans avoir défini ce qui était à protéger : en d'autres termes toute organisation désireuse de protéger ses systèmes et ses réseaux doit déterminer son *périmètre de sécurité*. Le périmètre de sécurité, au sein

de l'univers physique, délimite l'intérieur et l'extérieur, mais sa définition doit aussi englober (ou pas) les entités immatérielles qui peuplent les ordinateurs et les réseaux, essentiellement les logiciels et en particulier les systèmes d'exploitation.

Une fois fixé ce périmètre, il faut aussi élaborer une politique de sécurité, c'est-à-dire décider de ce qui est autorisé et de ce qui est interdit. À cette politique viennent bien sûr s'ajouter les lois et les règlements en vigueur, qui s'imposent à tous. Cela fait, il sera possible de mettre en place les solutions techniques appropriées à la défense du périmètre selon la politique choisie. Mais déjà il est patent que les dispositifs techniques ne pourront pas résoudre tous les problèmes de sécurité, et, de surcroît, la notion même de périmètre de sécurité est aujourd'hui battue en brèche par des phénomènes comme la multiplication des ordinateurs portables qui, par définition, se déplacent de l'intérieur à l'extérieur et inversement, à quoi s'ajoute l'extraterritorialité de fait des activités sur l'Internet.

Périmètre et frontière

La notion de périmètre de sécurité, ainsi que le signalait déjà l'alinéa précédent, devient de plus en plus fragile au fur et à mesure que les frontières entre l'extérieur et l'intérieur de l'entreprise ainsi qu'entre les pays deviennent plus floues et plus poreuses. Interviennent ici des considérations topographiques : les ordinateurs portables entrent et sortent des locaux et des réseaux internes pour aller se faire contaminer à l'extérieur ; mais aussi des considérations logiques : quelles sont les lois et les règles qui peuvent s'appliquer à un serveur hébergé aux États-Unis, qui appartient à une entreprise française et qui sert des clients brésiliens et canadiens ?

La justice et les fournisseurs français d'accès à l'Internet (FAI) en ont fait l'expérience : un certain nombre d'organisations ont déposé devant les tribunaux français des plaintes destinées à faire cesser la propagation de pages Web à contenus négationnistes, effectivement attaquables en droit français. Mais les sites négationnistes étaient installés aux États-Unis, pays dépourvu d'une législation anti-négationniste, ce qui interdisait tout recours contre les auteurs et les éditeurs des pages en question. Les plaignants se sont donc retournés contre les FAI français, par l'intermédiaire desquels les internautes pouvaient accéder aux pages délictueuses, mais ceux-ci n'en pouvaient mais. En effet, ainsi que nous le verrons à la page 234, le filtrage de contenus sur l'Internet est une entreprise coûteuse,

aux résultats incertains, et en fin de compte vaine, car les éditeurs des pages en question disposent de nombreux moyens pour déjouer les mesures de prohibition.

Cette question du filtrage de contenu est traitée par le rapport Kahn-Brugidou [25] ; le site `www.legalis.net` [73] assure une veille juridique bien faite sur toutes les questions liées aux développements de l'informatique et de l'Internet ; les livres de Solveig Godeluck [59] et de Lawrence Lessig [74] replacent ces questions dans un contexte plus général.

Ressources publiques, ressources privées

Les systèmes et les réseaux comportent des données et des programmes que nous considérerons comme des *ressources*. Certaines ressources sont d'accès public, ainsi certains serveurs Web, d'autres sont privées pour une personne, comme une boîte à lettres électronique, d'autres sont privées pour un groupe de personnes, comme l'annuaire téléphonique interne d'une entreprise. Ce caractère plus ou moins public d'une ressource doit être traduit dans le système sous forme de *droits d'accès*, comme nous le verrons à la page 37 où cette notion est présentée.

Identifier et authentifier

Les personnes qui accèdent à une ressource non publique doivent être *identifiées* ; leur identité doit être *authentifiée* ; leurs droits d'accès doivent être *vérifiés* au regard des *habilitations* qui leur ont été attribuées : à ces trois actions correspond un premier domaine des techniques de sécurité, les méthodes d'**authentification**, de signature, de vérification de l'**intégrité** des données et d'attribution de droits (une habilitation donnée à un utilisateur et consignée dans une base de données adéquate est une liste de droits d'accès et de pouvoirs formulés de telle sorte qu'un système informatique puisse les vérifier automatiquement).

La sécurité des accès par le réseau à une ressource protégée n'est pas suffisamment garantie par la seule identification de leurs auteurs. Sur un réseau local de type Ethernet où la circulation des données fonctionne selon le modèle de l'émission radiophonique que tout le monde peut capter (enfin, pas si facilement que cela, heureusement), il est possible à un tiers de la détourner. Si la transmission a lieu à travers l'Internet, les données circulent de façon analogue à une carte postale, c'est-à-dire qu'au moins le facteur et la concierge y ont accès. Dès lors que les données

doivent être protégées, il faut faire appel aux techniques d'un autre domaine de la sécurité informatique : le **chiffrement**.

Authentification et chiffrement sont indissociables : chiffrer sans authentifier ne protège pas des usurpations d'identité (comme notamment l'attaque par interposition, dite en anglais attaque de type *man in the middle*, et décrite à la page 50), authentifier sans chiffrer laisse la porte ouverte au vol de données.

Empêcher les intrusions

Mais ces deux méthodes de sécurité ne suffisent pas, il faut en outre se prémunir contre les intrusions destinées à détruire ou corrompre les données, ou à en rendre l'accès impossible. Les techniques classiques contre ce risque sont l'usage de **pare-feu** *(firewalls)* et le **filtrage** des communications réseaux, qui permettent de protéger la partie privée d'un réseau dont les stations pourront communiquer avec l'Internet sans en être « visibles » ; le terme *visible* est ici une métaphore qui exprime que nul système connecté à l'Internet ne peut de sa propre initiative accéder aux machines du réseau local (seules ces dernières peuvent établir un dialogue) et que le filtre interdit certains types de dialogues ou de services, ou certains correspondants (reconnus dangereux).

La plupart des entreprises mettent en place des ordinateurs qu'elles souhaitent rendre accessibles aux visiteurs extérieurs, tels que leur serveur Web et leur relais de messagerie. Entre le réseau privé et l'Internet, ces machines publiques seront placées sur un segment du réseau ouvert aux accès en provenance de l'extérieur, mais relativement isolé du réseau intérieur, afin qu'un visiteur étranger à l'entreprise ne puisse pas accéder aux machines à usage strictement privé. Un tel segment de réseau est appelé *zone démilitarisée* (**DMZ**), en souvenir de la zone du même nom qui a été établie entre les belligérants à la fin de la guerre de Corée. Les machines en DMZ, exposées donc au feu de l'Internet, seront appelées **bastions**.

Certains auteurs considèrent que ces techniques de sécurité par remparts, ponts-levis et échauguettes sont dignes du Moyen-Âge de l'informatique ; ils leur préfèrent les systèmes de détection d'intrusion (IDS), plus subtils, qui sont décrits par la page 230 et ses sous-sections. La surenchère suivante proclame que si l'on a détecté une intrusion, autant la stopper, et les IDS sont devenus des IPS (systèmes de prévention d'intrusion). Et l'on verra plus loin que les IPS sont critiqués par les tenants des mandataires applicatifs, plus subtils encore. Cela dit, dans un pay-

sage informatique où les micro-ordinateurs prolifèrent sans qu'il soit réaliste de prétendre vérifier la configuration de chacun, le filtrage et le pare-feu sont encore irremplaçables.

Pour couper court à toutes ces querelles autour des qualités respectives de telle ou telle méthode de sécurité, il suffit d'observer l'état actuel des menaces et des vulnérabilités. Il y a encore une dizaine d'années, le paramétrage de filtres judicieux sur le routeur de sortie du réseau d'une entreprise vers l'Internet pouvait être considéré comme une mesure de sécurité bien suffisante à toutes fins pratiques. Puis il a fallu déployer des antivirus sur les postes de travail. Aujourd'hui, les CERT (*Computer Emergency Response Teams*, voir page 17 pour une description de ces centres de diffusion d'informations de sécurité informatique) publient une dizaine de vulnérabilités nouvelles par semaine, et l'idée de pouvoir se prémunir en flux tendu contre toutes est utopique. La conception moderne (en cette année 2006) de la protection des systèmes et des réseaux s'appuie sur la notion de *défense en profondeur*, par opposition à la défense frontale rigide, où l'on mise tout sur l'efficacité absolue d'un dispositif unique.

Défense en profondeur

La défense en profondeur —au sujet de laquelle on lira avec profit un article du Général Bailey [12] qui évoque à son propos une véritable « révolution dans les affaires militaires » — consiste à envisager que l'ennemi puisse franchir une ligne de défense sans pour cela qu'il devienne impossible de l'arrêter ; cette conception s'impose dès lors que les moyens de frappe à distance et de déplacement rapide, ainsi que le combat dans les trois dimensions, amènent à relativiser la notion de ligne de front et à concevoir l'affrontement armé sur un territoire étendu. Plus modestement, la multiplication des vulnérabilités, la généralisation des ordinateurs portables qui se déplacent hors du réseau de l'entreprise, l'usage de logiciels novateurs (code mobile, *peer to peer*, sites interactifs, téléphonie et visioconférence sur IP) et d'autres innovations ont anéanti la notion de « périmètre de sécurité » de l'entreprise, et obligent le responsable SSI à considérer que la menace est partout et peut se manifester n'importe où. Il faut continuer à essayer d'empêcher les intrusions dans le SI de l'entreprise, mais le succès de la prévention ne peut plus être garanti, et il faut donc se préparer à limiter les conséquences d'une attaque réussie, qui se produira forcément un jour. Et ce d'autant plus que le SI contemporain n'est pas comme par le passé contenu par un « centre de données » monolithique

hébergé dans un bunker, mais constitué de multiples éléments plus ou moins immatériels qui vivent sur des ordinateurs multiples, dispersés dans toute l'entreprise et au dehors ; et c'est cette nébuleuse qu'il faut protéger.

Nous allons au cours des chapitres suivants examiner un peu plus en détail certaines collections de techniques qui s'offrent au responsable SSI, en commençant par la cryptographie dont sont dérivées les techniques de l'authentification.

Aspects organisationnels de la sécurité

À côté des mesures techniques destinées à assurer la protection des systèmes et des réseaux, la sécurité du SI comporte un volet humain et social au moins aussi important : la sécurité dépend en dernière analyse des comportements humains et, si les comportements sont inadaptés, toutes les mesures techniques seront parfaitement vaines parce que contournées.

Abandonner les utilisateurs inexpérimentés aux requins ?

Un article récent de Marcus J. Ranum [88] (voir aussi page 224), qui n'est rien moins que l'inventeur du pare-feu et une autorité mondiale du domaine SSI, soutient l'idée paradoxale qu'il serait inutile, voire nuisible, d'éduquer les utilisateurs du SI à la sécurité : son argument est que les utilisateurs incapables de maîtriser suffisamment leur ordinateur, notamment en termes de mesures de sécurité, sont condamnés à être expulsés du marché du travail, et qu'il ne faut rien faire pour les sauver. Cette idée ne peut manquer de séduire les RSSI épuisés non pas tant par l'inconscience et l'ignorance de leurs utilisateurs, que par le fait que ceux-ci *ne veulent rien savoir*. Cela dit, après avoir jubilé quelques instants à l'idée de la disparition en masse de ses utilisateurs les plus insupportables, le RSSI se retrouve par la pensée dans la situation du narrateur d'un récit de Roland Topor [111], naufragé reçu comme dieu vivant d'une île du Pacifique, et qui un jour, exaspéré par une rage de dents, crie à ses fidèles « Vous pouvez tous crever ! », suggestion à laquelle ils obéissent incontinent.

Si la suggestion de M. Ranum n'est pas à prendre à la légère, il convient néanmoins de prendre en considération que les questions de SSI sont fort complexes et évoluent vite, si bien que même les utilisateurs avertis peuvent être pris de court

par des menaces dont ils n'étaient pas informés. Nous pouvons même risquer une assertion plus générale : en informatique, *aucune compétence n'est pérenne ni complète*. Il convient donc que les RSSI et de façon plus générale tous les informaticiens responsables des infrastructures techniques et des réseaux consacrent une part de leur activité à informer, sensibiliser et former les utilisateurs à la problématique SSI. Eux-mêmes doivent se tenir en permanence au courant de l'évolution du sujet, être abonnés aux bulletins d'alerte des CERT et aux revues spécialisées, fréquenter les forums spécialisés et les conférences et mettre en application les enseignements qu'ils en auront tirés. Tout cela semblerait aller de soi, si l'on ne voyait combien peu ces conseils sont entendus.

Idéalement, dans une entreprise, aucun utilisateur ne devrait être laissé « à l'abandon », c'est-à-dire avec un accès incontrôlé au réseau de l'entreprise et à ses communications avec l'Internet, il devrait y avoir dans chaque groupe de travail un correspondant informatique en contact avec les responsables des infrastructures et du réseau. En l'absence d'une telle structure d'échanges, les phénomènes les plus dangereux ne manqueront pas de proliférer, et de ce moment surgiront les incidents les plus graves.

La nature du « contact » entre le correspondant informatique et les responsables du SI et des infrastructures pourra dépendre du type d'organisation : dans une entreprise assez centralisée et hiérarchisée la fonction de correspondant informatique sera définie en termes opérationnels, il aura des directives précises à appliquer et devra rendre compte de leur application ainsi que de tout problème informatique qui pourrait survenir. Dans une entreprise à la structure plus lâche, un organisme de recherche par exemple, la mise en place d'une telle organisation peut se révéler difficile, les relations de contact seront moins formelles, mais il sera néanmoins important qu'elles existent, ne serait-ce que par des conversations régulières au pied de la machine à café.

Externalisation radicale ?

En septembre 2004 un article de *Computer Weekly* [97] a signalé une politique d'une nouveauté bouleversante pour faire face à la dissolution du périmètre de sécurité (on parle désormais de *dépérimétrisation*). *British Petroleum (BP)*, la firme pétrolière bien connue, était obligée d'administrer 380 extranets pour communiquer avec 90 000 correspondants d'entreprises clients, fournisseurs ou partenaires de

par le monde, et ce au travers des infrastructures infiniment variées en nature et en qualité des opérateurs locaux. Elle a décidé qu'il serait beaucoup plus simple et efficace de leur offrir, par l'Internet, un accès analogue à celui que les banques offrent à leurs clients pour gérer leur compte.

La démarche ne s'est pas arrêtée là : BP s'est rendu compte que cette solution d'accès pourrait être étendue à une fraction de son propre personnel, estimée à 60% de ses 96 200 employés, qui n'avaient pas besoin d'utiliser de systèmes client-serveur particuliers, un navigateur suffirait.

Les avantages d'une telle solution semblent considérables : l'entreprise n'a plus besoin de se soucier de la sécurité sur le poste de travail des correspondants ou des employés ainsi « externalisés », pas plus que la banque ne s'occupe de l'ordinateur de son client. C'est leur problème.

Sauvegarder données et documents

La sauvegarde régulière des données et de la documentation qui permet de les utiliser est bien sûr un élément indispensable de la sécurité du système d'information, elle constitue un sujet d'étude à elle seule, qui justifierait un livre entier. Aussi ne ferons-nous, dans le cadre du présent ouvrage, que l'évoquer brièvement, sans aborder les aspects techniques. Mentionnons ici quelques règles de bon sens :

- Pour chaque ensemble de données il convient de déterminer la périodicité des opérations de sauvegarde en fonction des nécessités liées au fonctionnement de l'entreprise.
- Les supports de sauvegarde doivent être stockés de façon à être disponibles après un sinistre tel qu'incendie ou inondation : armoires ignifugées étanches ou site externe.
- Les techniques modernes de stockage des données, telles que *Storage Area Network* (SAN) ou *Network Attached Storage* (NAS), conjuguées à la disponibilité de réseaux à haut débit, permettent la duplication de données à distance de plusieurs kilomètres (voire plus si l'obstacle financier n'est pas à considérer), et ce éventuellement en temps réel ou à intervalles très rapprochés ; ce type de solution est idéal pour un site de secours.
- De l'alinéa précédent, on déduit que, dans un système d'information moderne, toutes les données doivent être stockées sur des SAN ou des NAS,

rien ne justifie l'usage des disques attachés directement aux serveurs, qui seront réservés aux systèmes d'exploitation et aux données de petit volume.

- Les dispositifs et les procédures de sauvegarde et, surtout, de restauration doivent être vérifiés régulièrement (cf. la section suivante).

Vérifier les dispositifs de sécurité

Le dispositif de sécurité le mieux conçu ne remplit son rôle que s'il est opérationnel, et surtout si ceux qui doivent, en cas de sinistre par exemple, le mettre en œuvre, sont eux aussi opérationnels. Il convient donc de vérifier régulièrement les capacités des dispositifs matériels et organisationnels.

Les incidents graves de sécurité ne surviennent heureusement pas tous les jours : de ce fait, si l'on attend qu'un tel événement survienne pour tester les procédures palliatives, elles risquent fort de se révéler défaillantes. Elles devront donc être exécutées « à blanc » périodiquement, par exemple en effectuant la restauration d'un ensemble de données à partir des sauvegardes tous les six mois, ou le redémarrage d'une application à partir du site de sauvegarde.

Outre ces vérifications régulières, l'organisation d'exercices qui simulent un événement de sécurité impromptu peut être très profitable. De tels exercices, inspirés des manœuvres militaires, révéleront des failles organisationnelles telles que rupture de la chaîne de commandement ou du circuit d'information. Un rythme bisannuel semble raisonnable pour ces opérations.

S'informer auprès des CERT

Les CERT *(Computer Emergency Response Teams)* centralisent, vérifient et publient les alertes relatives à la sécurité des ordinateurs, et notamment les annonces de vulnérabilités récemment découvertes. Les alertes peuvent émaner des auteurs du logiciel, ou d'utilisateurs qui ont détecté le problème. Détecter une vulnérabilité ne veut pas dire qu'elle soit exploitée, ni même exploitable, mais le risque existe.

Organisation des CERT

Les vulnérabilités publiées par les CERT sont relatives à toutes sortes de systèmes ; leur publication constitue une incitation forte pour que les industriels concernés (les producteurs du système ou du logiciel le plus souvent) les corrigent. Certains tentent aussi de ralentir le travail des CERT, dont ils aimeraient bien qu'ils ne dévoilent pas leurs faiblesses.

Le premier CERT a vu le jour à l'université Carnegie-Mellon de Pittsburgh en novembre 1988, sur une initiative de la DARPA *(Defense Advanced Research Projects Agency)* consécutive à la propagation du ver de Morris, la première attaque, involontaire[1] mais de grande envergure, contre l'Internet. En 2006 la France dispose de trois CERT : le CERTA[2] pour les besoins des administrations et services publics, le CERT Renater[3] qui s'adresse aux universités et centres de recherche, le CERT-IST[4] qui s'adresse au monde industriel. En fait la coopération au sein de la communauté mondiale des CERT est assez étroite, surtout en période de crise. Cette communauté est concrétisée par l'existence d'un Centre de Coordination des CERT[5], hébergé par l'université Carnegie Mellon à Pittsburgh en Pennsylvanie.

La publication des avis des CERT est une contribution majeure et vitale à la sécurité des systèmes d'information. Leur volume est tel que le dépouillement, qui ne peut être confié qu'à des ingénieurs réseau de haut niveau, représente un travail considérable.

Faut-il publier les failles de sécurité ?

Un débat s'est engagé sur le bien-fondé de certains avis, et sur la relation qu'il pourrait y avoir entre le nombre d'avis concernant un logiciel ou un système donné et sa qualité intrinsèque. Les détracteurs des logiciels libres ont mis en exergue le volume très important d'avis des CERT qui concernaient ceux-ci (par exemple Linux, le serveur Web *Apache*, *Sendmail*, etc.) pour en inférer leur fragilité. Leurs défenseurs ont riposté en expliquant que les avis des CERT concernaient par dé-

[1] Du moins à en croire son auteur Robert Tappan Morris.
[2] Cf. `http://www.certa.ssi.gouv.fr/`
[3] Cf. `http://www.renater.fr/rubrique.php3?id_rubrique=19`
[4] Cf. `http://www.cert-ist.com/`
[5] Cf. `http://www.cert.org/`

finition des failles de sécurité découvertes et donc virtuellement corrigées, alors que l'absence d'avis relatifs à tel système commercial pouvait simplement signifier que l'on passait sous silence ses défauts de sécurité en profitant de son opacité. Or l'expérience montre que tout dispositif de sécurité a des failles ; les attaquants ne perdent pas leur temps à faire de la recherche fondamentale sur la factorisation des grands nombres entiers, ils essaient de repérer les failles d'implémentation et ils les exploitent. Face à ce risque, la meilleure protection est une capacité de riposte rapide, qui consiste le plus souvent à commencer par désactiver le composant pris en défaut en attendant la correction. La communauté du logiciel libre excelle dans cet exercice, mais avec les logiciels commerciaux les utilisateurs n'ont souvent aucun moyen d'agir : ils ne peuvent qu'attendre le bon vouloir de leur fournisseur. Dans ce contexte, la publication d'avis des CERT relatifs à des logiciels commerciaux est très bénéfique parce qu'elle incite les fournisseurs à corriger plus rapidement un défaut dont la notoriété risque de nuire à leur réputation. Mais certains fournisseurs cherchent à obtenir le silence des CERT en arguant du fait que leurs avis risquent de donner aux pirates des indications précieuses… ce qui est fallacieux car les sites Web des pirates sont de toute façon très bien informés et mis à jour, eux, selon les principes du logiciel libre, ce qui indique bien où est l'efficacité maximale. L'expérience tend à prouver qu'une faille de sécurité est d'autant plus vite comblée qu'elle est publiée tôt et largement. L'accès au code source du logiciel en défaut constitue bien sûr un atout.

La réponse à la question posée par le titre de cette section est donc : *oui, il faut publier les failles de sécurité, mais de façon organisée et responsable*, c'est-à-dire de façon certifiée, sur le site d'un organisme accrédité, typiquement un CERT, et après avoir prévenu l'auteur ou l'éditeur du logiciel en défaut et lui avoir laissé un délai raisonnable pour au moins trouver un palliatif d'urgence. Il faut savoir qu'il existe aujourd'hui un marché de la faille, qui parfois n'est pas loin de s'apparenter à du chantage.

Le management de la sécurité

Cette section doit beaucoup à la formation ISO 27001 Lead Auditor, *dispensée par Alexandre Fernandez et Hervé Schauer, de Hervé Schauer Consultants. L'un d'entre nous a suivi cette formation et obtenu la certification à laquelle elle prépare. Qu'A. Fernandez et H. Schauer soient ici remerciés pour avoir su rendre captivante une matière plutôt aride. Les erreurs et imprécisions ne peuvent être imputées qu'à l'auteur.*

La présente section sur le management de la sécurité présente des normes et des méthodes que nous n'approuvons pas ; elles ne sont pas inutiles, mais nuisibles. Néanmoins il convient qu'elles aient leur place dans ce livre, d'abord parce que sans elles cet exposé serait incomplet, ensuite parce que tout responsable de la sécurité a intérêt à les connaître s'il veut conserver son emploi. Nous ne saurions trop recommander au responsable sécurité soucieux de son avenir professionnel de suivre une formation du type de celle qui est mentionnée en exergue de cette section.

CULTURE « Management », un faux anglicisme

Pour se résigner à l'emploi du mot management on se rappellera que, loin d'être un anglicisme, il s'agit d'un vieux mot français remis à l'honneur : Olivier de Serres (1539-1619) emploie en effet le terme *ménager* dans une acception qui en fait le *manager* contemporain, on nous fera grâce de la variation orthographique, courante à l'époque. Et l'emploi du mot *gestion* à toutes les sauces serait bien pire.

Les systèmes de management

L'Organisation internationale de normalisation, ou *International organization for standardization* en anglais (ISO pour la forme abrégée) est une organisation internationale, créée en 1947, composée de représentants des organismes de normalisation nationaux d'environ 150 pays, qui produit des normes internationales dans des domaines industriels et commerciaux.

L'ISO a entrepris d'encadrer par des normes les *systèmes de management*, et pour ce faire a commencé par en donner une définition, qui fait l'objet de la norme IS (pour *International Standard*) 9000 ; un système de management est un système qui permet :

- d'établir une politique ;
- de fixer des objectifs ;
- de vérifier que l'on a atteint les objectifs fixés.

Plus concrètement, un système de management comporte un ensemble de mesures organisationnelles et techniques destinées à mettre en place un certain contexte organisationnel et à en assurer la pérennité et l'amélioration. L'idée cruciale au cœur de cette problématique est que le système de management repose sur un référentiel écrit, et qu'il est donc *vérifiable*, au moyen d'un *audit* qui consistera à comparer le référentiel à la réalité pour relever les divergences, nommées *écarts* ou *non-conformités*.

Il existe actuellement (en 2006) trois normes relatives aux systèmes de management :

- la norme IS 9001 consacrée aux systèmes de management de la qualité et aux exigences associées ;
- la norme IS 14001 consacrée aux systèmes de management de l'environnement ;
- la norme IS 27001 consacrée aux systèmes de management de la sécurité de l'information ; c'est cette dernière qui nous intéressera plus particulièrement ici.

Pour couronner cet édifice remarquable, la norme IS 19001 formule les directives à respecter pour la conduite de l'audit d'un système de management.

Le système de management de la sécurité de l'information

La norme IS 27001 [68] est destinée à s'appliquer à un système de management de la sécurité de l'information (SMSI) ; elle comporte notamment un schéma de certification susceptible d'être appliqué au SMSI au moyen d'un audit.

Comme toutes les normes relatives aux systèmes de management, IS 27001 repose sur une approche par *processus*, et plus précisément sur le modèle de processus formulé par W.E. Deming, du MIT, et nommé *roue de Deming*, ou PDCA, comme *Plan, Do, Check, Act* :

- phase *Plan* : définir le champ du SMSI, identifier et évaluer les risques, produire le document *(Statement of applicability, SOA)* qui énumère les *mesures de sécurité* à appliquer ;

- phase *Do* : affecter les ressources nécessaires, rédiger la documentation, former le personnel, appliquer les mesures décidées, identifier les risques résiduels ;
- phase *Check* : audit et revue périodiques du SMSI, qui produisent des *constats* et permettent d'imaginer des corrections et des améliorations ;
- phase *Act* : prendre les mesures qui permettent de réaliser les corrections et les améliorations dont l'opportunité a été mise en lumière par la phase *Check*, préparer une nouvelle itération de la phase *Plan*.

Le SMSI a pour but de maintenir et d'améliorer la position de l'organisme qui le met en œuvre du point de vue, selon les cas, de la compétitivité, de la profitabilité, de la conformité aux lois et aux règlements, et de l'image de marque. Pour cela il doit contribuer à protéger les actifs *(assets)* de l'organisme, définis au sens large comme tout ce qui compte pour lui.

Pour déterminer les mesures de sécurité dont la phase *Plan* devra fournir une énumération, la norme IS 27001 s'appuie sur le catalogue de mesures et de bonnes pratiques proposé par la norme IS 17799, « *International Security Standard* » [69], plus volumineuse et au contenu plus technique.

IS 27001 impose une analyse des risques, mais ne propose aucune méthode pour la réaliser : l'auteur du SMSI est libre de choisir la méthode qui lui convient, à condition qu'elle soit documentée et qu'elle garantisse que les évaluations réalisées avec son aide produisent des résultats comparables et reproductibles. Un risque peut être accepté, transféré à un tiers (assurance, prestataire), ou réduit à un niveau accepté.

Un exemple de méthode d'analyse de risque utilisable dans le cadre d'IS 27001 est la méthode EBIOS® (Expression des Besoins et Identification des Objectifs de Sécurité) [6], qui « permet d'apprécier et de traiter les risques relatifs à la sécurité des systèmes d'information (SSI). Elle permet aussi de communiquer à leur sujet au sein de l'organisme et vis-à-vis de ses partenaires afin de contribuer au processus de gestion des risques SSI. »

[6] `http://www.ssi.gouv.fr/fr/confiance/ebiospresentation.html`.

Élaboration et mise en place du SMSI

La norme IS 27001 précise la démarche qui doit être suivie pour élaborer et mettre en place le SMSI : sans entrer trop dans les détails, ce qui risquerait d'enfreindre les droits des organismes de normalisation qui vendent fort cher les textes des normes, disons que l'organisme désireux de se voir certifier devra :

- définir le champ du SMSI ;
- en formuler la politique de management ;
- préciser la méthode d'analyse de risques utilisée ;
- identifier, analyser et évaluer les risques ;
- déterminer les traitements qui seront appliqués aux différents risques, ainsi que les moyens d'en vérifier les effets ;
- attester l'engagement de la direction de l'organisme dans la démarche du SMSI ;
- rédiger le *Statement of Applicability* (SOA) qui sera la charte du SMSI et qui permettra de le soumettre à un audit.

Suivi et application du SMSI

Ici la norme précise que, une fois que le SMSI a été formulé, il faut faire ce qu'il stipule, vérifier que c'est fait, identifier les erreurs dans son application, les failles qui s'y manifestent, les modifications du contexte de nature à nécessiter sa mise à jour ou sa modification.

Tâches de direction et d'encadrement

À la direction de l'organisme, dont il a déjà été dit qu'elle devait s'engager activement dans la démarche, incombent d'autres obligations : vérifier que tout est bien fait selon les règles, affecter à la démarche du SMSI des ressources suffisantes en personnel et en moyens matériels, déterminer les besoins qui en résultent en termes de compétence et de formation, fournir les efforts qui conviennent en termes de sensibilisation et de formation, effectuer le contrôle des effets de ces efforts. Il faut aussi organiser des revues et des exercices, etc., tout cela afin d'assurer l'*amélioration continue* du SMSI. Cette vision idyllique d'un univers en marche vers le Bien, le Beau, le Juste ne saurait manquer de soulever l'enthousiasme du lecteur !

Un modèle de maturité ?

La norme ISO/IEC 21827 [65] propose un « Modèle de maturité de capacité » : qui peut traduire ce jargon invraisemblable ?

Critères communs

Les *Critères Communs* (norme ISO/IEC 15408) sont étrangers ou plutôt parallèles à la démarche IS 27001 ; ils se proposent de servir de base pour l'évaluation des propriétés de sécurité des produits et des systèmes de traitement de l'information. Nous n'en dirons guère plus ici, parce que cette norme s'adresse aux concepteurs de produits de sécurité plutôt qu'à ceux qui les utilisent pour construire des systèmes d'information sûrs.

Faut-il adhérer aux normes de sécurité de l'information ?

L'auteur de ces lignes n'est pas convaincu que les normes évoquées à la section précédente soient un remède à l'insécurité ; ces méthodes sont d'une grande lourdeur, leur seul apprentissage est d'une ampleur propre à absorber une énergie considérable, or une fois que l'on connaît par cœur les critères communs et que l'on sait appliquer EBIOS les pieds au mur, on n'a pas mis en place une seule mesure concrète de SSI, on est seulement capable, en principe, d'évaluer les mesures que d'autres auront éventuellement mises en place.

Ce qui frappe le lecteur, c'est que la vérification formelle de conformité à ces normes peut presque être effectuée par un auditeur dépourvu de compétence technique : il suffit de lire les documents obligatoires et de vérifier que les mesures mentionnées ont bien été appliquées, ce qui doit être écrit dans un autre document. On pourrait presque imaginer un audit par ordinateur : il serait sans doute mauvais, mais formellement conforme. Reste à écrire le compilateur de normes et le compilateur de SOA. Évidemment, pour réaliser un *bon* audit, l'intuition de l'auditeur, nourrie par son expérience, jouera un rôle important. Certains collègues dont je tairai les noms de crainte de leur attirer des ennuis vont jusqu'à dire que l'adoption d'une démarche telle que celle proposée par IS 27001 ou IS 21827 est nuisible, elle empêcherait les gens de penser correctement, de se poser les bonnes questions. Si j'osais je serais d'accord avec eux, mais je suis trop respectueux des normes et des autorités pour cela. En fait, les auteurs de ces normes semblent croire que l'univers peut être décrit de façon adéquate par un tableau de cases à

cocher, analogue à un questionnaire à choix multiples : on se demande pourquoi les grands nigauds nommés Aristote, Descartes, Newton, Kant et Einstein n'y ont pas pensé, ils se seraient épargné bien de la fatigue cérébrale.

Une autre faiblesse de ces démarches, c'est leur déterminisme : la lecture de leurs documentations suggère que l'univers des risques et des menaces qu'elles sont censées conjurer est parfaitement ordonné et prévisible, alors que justement ses caractéristiques premières sont le chaos et la surprise. De ce fait, le temps passé à cocher consciencieusement les cases du tableau *Excel* où l'on aura reporté les rubriques de son SOA *(Statement of Applicability)* risque d'avoir été perdu, et il aurait sans doute été plus judicieux de le consacrer à de la sécurité réelle. Soulignons à cette occasion les ravages exercés par un logiciel autrement bien pratique, *Excel* : pour certains managers, le monde semble pouvoir être décrit par un tableau de cases ; dès qu'un problème a plus de deux dimensions, c'est la panique parce que cela n'entre plus dans le tableur.

Une telle vision, malgré sa pauvreté, comporte une métaphysique implicite, dont Isabelle Boydens [22] donne un énoncé explicite (p. 62) :

« Une telle approche repose implicitement sur trois postulats : »

- « Le monde est composé d'éléments discrets, univoques, clairement identifiables et perceptibles.
- Les combinaisons et la connaissance de ces éléments sont gouvernées par des lois.
- Il est possible d'établir une relation bi-univoque entre le réel observable et sa représentation informatique en vertu de l'isomorphisme qui les relierait l'un à l'autre. »

Bien sûr, le monde n'est pas ainsi. Cela dit il ne convient pas d'ignorer que, dans les grandes structures bureaucratisées, ce type de démarche est devenu à peu près inévitable, un peu comme ISO 9001. Les procédures destinées à évaluer des travaux techniques deviennent une charge de travail plus lourde que l'objet de l'évaluation, les procédures de gestion demandent plus de travail que les activités qu'elles servent à gérer, bref ce qui devrait être une aide pour l'action devient un fardeau, de surcroît ennuyeux.

Pour résumer cette analyse en une formule : toutes ces normes et ces procédures n'ont qu'une finalité, permettre à des incompétents de diriger.

Un autre défaut de ces procédures d'évaluation, c'est qu'elles ne sont pas uniquement construites en fonction des buts à atteindre, mais aussi, sinon surtout, en fonction de ce qui, dans les processus étudiés, se prête bien à l'évaluation, parce que par exemple il est facile d'y adapter une métrique. Conformément au proverbe, pour celui qui ne dispose que d'un marteau, tout ressemble à un clou, et les normalisateurs de la sécurité n'ont pas toujours échappé à ce travers.

Le RSSI qui aura pu échapper à la lourdeur de ces carcans normalisés aura à cœur d'élaborer une politique et des règles de sécurité raisonnables, sobres, les plus simples possible, et adaptées à la situation locale. Le présent ouvrage se veut un guide pour rédiger une politique de sécurité[7].

De toutes les façons il faut savoir que des règles de sécurité complexes ou trop contraignantes seront simplement inappliquées, parce que trop difficiles à comprendre. La simple lecture des critères communs et des manuels EBIOS représente des milliers de pages : autant dire que leur étude détaillée est antinomique de toute politique réelle de sécurité. Leur fonction principale ne serait-elle pas de donner du travail aux cabinets de consultants spécialisés, à condition que leur clientèle soit (très) solvable ?

Législation financière et système d'information

Les questions techniques et organisationnelles ne sont pas les seules à avoir des effets sur la sécurité du système d'information. Après le management de la sécurité et ses excès, nous aborderons ici l'application de la sécurité au management, qui engendre elle aussi des pratiques abusives.

L'ubiquité de l'informatique est telle que des mesures législatives destinées à réglementer des domaines que l'on pourrait croire très éloignés de l'objet du présent ouvrage finissent par se trouver au cœur de sa problématique. Un de nos collègues distinguait la « sécurité dure » (crypto-processeurs, pare-feu, réseaux privés virtuels, séparation des privilèges) de la « sécurité molle », qui par analogie avec les sciences affublées du même adjectif se préoccupe de ces aspects simplement humains : ce sont de certains d'entre eux qu'il sera question ici. L'administrateur de système et de réseau pourrait se croire à l'abri des monstres bureaucratiques men-

[7] Le lecteur pourra aussi se reporter avec profit au livre bénéfiquement concis de Scott Barman. *Writing Information Security Policies*. New Riders, Indianapolis, USA, 2002.

tionnés ci-dessous : qu'il s'estime heureux si on ne lui impose pas les procédures éléphantesques qu'ils engendrent.

Législation financière et SI

Depuis les scandales financiers de la période 2001-2002 (nous ne mentionnerons ici que les affaires Enron et Worldcom), sont apparues comme champignons après la pluie des réglementations destinées à améliorer le contrôle des autorités et des actionnaires sur la gestion des entreprises. Le signal a bien sûr été donné par les États-Unis en juillet 2002 avec la loi Sarbanes-Oxley (plus familièrement SOX), qui impose aux entreprises qui font appel au capital public (c'est-à-dire cotées en bourse) toute une série de règles comptables et administratives destinées à assurer la traçabilité de leurs opérations financières, afin que les actionnaires ne courent plus le risque de voir leurs actions partir en fumée après une déconfiture que des comptes truqués n'auraient pas permis de prévoir, cependant que les dirigeants initiés auraient revendu à temps leurs stock-options pour se retirer sur leur yacht aux îles Cayman... La France a bien sûr emboîté le pas avec la loi du 1er août 2003 sur la sécurité financière (LSF) qui concerne principalement trois domaines : la modernisation des autorités de contrôle des marchés financiers, la sécurité des épargnants et des assurés et enfin le contrôle légal des comptes ainsi que la transparence et le gouvernement d'entreprise. Cette loi française ne concerne pas seulement les sociétés cotées, mais toutes les sociétés anonymes ; elle est complétée par le dispositif réglementaire européen « Bâle 2 » de 2004, qui concerne les établissements financiers.

La conséquence pratique la plus visible de ces législations, c'est la prolifération des systèmes de contrôle et d'audit que nous avons évoqués à la page 20, et c'est bien pourquoi le responsable de sécurité ne peut les ignorer.

La loi Sarbanes-Oxley concerne la sécurité du système d'information en ceci qu'elle impose aux entreprises des procédures de contrôle interne, de conservation des informations, et de garantie de leur exactitude. La description détaillée de ces procédures, et de leur réalisation dans le système d'information, est un élément clé de la loi, notamment pour ce qui a trait aux points suivants :

1. la continuité des opérations ;
2. la sauvegarde et l'archivage des données ;
3. l'externalisation et son contrôle.

Les législations européennes ont emprunté les mêmes chemins.

Brève critique de la sécurité financière

On peut lire sur le site de *VLSI Research* un article [64] dans lequel son président G. Dan Hutcheson fait une analyse très pessimiste des perspectives de l'économie américaine postérieures à l'affaire Enron et à la floraison de ces législations.

Hutcheson retient les points suivants :

1. la quasi-disparition des stock-options prive les entreprises émergentes du moyen de motiver leur personnel ;
2. la lourdeur et le coût considérables de l'adaptation à la loi Sarbanes-Oxley empêcheront pratiquement les entreprises émergentes d'entrer en bourse, c'est-à-dire d'accéder aux sources de capital (notons que les éventuelles entreprises émergentes françaises n'auront pas à souffrir un tel dommage, puisque l'accès au marché boursier leur est déjà pratiquement impossible) ;
3. cette fermeture du marché boursier aux entreprises émergentes casse le modèle américain de capital-risque, sur lequel reposait la créativité industrielle du pays ;
4. les analystes financiers optimistes, accusés d'entraîner les épargnants dans des aventures dangereuses, risquent désormais la prison : on peut s'attendre à une flambée de pessimisme ;
5. l'orientation des entreprises selon les nouveaux impératifs de réduction des coûts et d'optimisation des achats coupe court à toute tentation d'innover.

Hutcheson est d'autant plus sévère à l'égard de la nouvelle législation que, selon lui, les lois existantes étaient tout à fait suffisantes pour assurer la transparence et lutter contre la fraude.

Ajoutons que ces différentes législations souffrent, selon nous, d'un vice de conception : elles suggèrent que la comptabilité des entreprises pourrait résulter de l'observation neutre et objective de phénomènes naturels, un peu comme les sciences de la nature, alors qu'un système comptable est construit selon des objectifs et des intentions. La comptabilité des entreprises est construite de façon à limiter l'exposition à la fiscalité, ce qui est un impératif autrement vital que la transparence économique ; quant à la comptabilité des organismes publics, en France tout au moins, elle essaye de se couler dans un carcan réglementaire dont les premières planches ont été clouées au XIVe siècle (cf. mon livre [19], ou sur le Web [20]).

La sécurité procédurale n'est pas la solution

Après ce tour d'horizon des normes de sécurité basées sur des procédures administratives et des excès de la sécurité appliquée au management, nous évoquerons les analyses de Jean-Pierre Dupuy [45], qui jettent une lumière vive aussi bien sur toutes ces normes relatives aux systèmes de management que sur la mode récente du principe de précaution.

Pour décrire ces systèmes de pensée, Dupuy introduit la notion de « rationalité procédurale », qui procéderait de réunions de comités d'experts, éventuellement à l'écoute de la société civile, et qui serait la forme consensuelle de la démocratie contemporaine. Ce modèle peut facilement être transposé à la gestion des entreprises, notamment par les méthodes de conduite de projet. « Dire que la rationalité est procédurale, c'est dire qu'une fois l'accord réalisé sur les justes et bonnes procédures, ce qu'elles produiront sera *ipso facto*, par propriété héritée en quelque sorte, juste et bon. C'est donc renoncer à chercher, indépendamment de et antérieurement à toute procédure, les critères du juste et du bien... » [nous pourrions ajouter : du vrai].

Les normes de systèmes de management (IS 9001 pour le management de la qualité, 14001 pour l'environnement, 27001 pour la sécurité de l'information) sont des outils à produire de la rationalité procédurale. Les normalisateurs eux-mêmes le revendiquent : disposer d'une organisation certifiée IS 9001 ne prouve en rien que l'organisation soit d'une qualité particulièrement excellente, cela signifie uniquement que les règles de fonctionnement de cette organisation sont documentées conformément à la norme (qui impose des règles dans certains domaines précis), et que des procédures existent pour vérifier que les règles sont appliquées, mais l'objet de ces procédures n'est en aucun cas de chercher à savoir si les décisions qui ont engendré ces règles étaient judicieuses. On peut dire la même chose des normes IS 14001 et 27001, chacune dans son domaine.

Pour continuer avec Dupuy : « La rationalité procédurale a du bon, sauf lorsqu'elle se construit au prix du renoncement à toute rationalité substantielle. » La sociologie des entreprises et l'évolution des rapports de pouvoir au sein des organisations techniques telles que les directions des systèmes d'information des entreprises, que j'ai décrites dans un ouvrage précédent [19], donnent à penser que c'est bien au renoncement à toute rationalité substantielle que conduisent les normes de système de management IS 9001 et IS 27001. En effet, pour un dirigeant paresseux, la

grande supériorité de la rationalité procédurale sur sa cousine substantielle, c'est qu'elle dispense de toute compétence sur son objet, et surtout de toute compétence technique, ce qui dans notre beau pays est une vertu cardinale, tant la compétence technique y est méprisée. Grâce aux systèmes de management, de simples cadres administratifs pourront exercer le pouvoir sur des ingénieurs compétents, puisqu'il leur suffira pour cela de cocher dans un tableur les cases qui correspondent aux étapes des procédures, et de prendre en défaut les acteurs opérationnels qui n'auront pas rempli toutes les cases, cependant qu'eux-mêmes ne seront bien sûr jamais exposés à telle mésaventure. Une caractéristique aussi attrayante rend inévitable le triomphe de ces normes, d'autant plus que la lourdeur des opérations de constitution des feuilles de tableur et de cochage des cases (il existe aussi un marché lucratif de logiciels spécialisés) permettra le développement démographique de la caste administrative et le renforcement de son hégémonie, sans oublier l'essor des cabinets spécialisés qui pourront vendre à prix d'or la mise en place de ces systèmes, puis la rédaction de rapports vides de tout contenu « substantiel ».

Il peut sembler hasardeux de formuler un jugement aussi négatif sur les méthodes désormais classiques de conduite de projet et sur les normes de système de management : si pratiquement tous les directeurs de système d'information les adoptent, c'est qu'il doit y avoir de bonnes raisons à cela, qu'ils doivent y trouver des avantages.

La réponse tient en trois points :

- Le succès de ces méthodes de management et des normes qui les accompagnent n'est pas uniforme à l'échelle internationale, par exemple l'engouement pour IS 9001 semble bien être une spécificité française.
- Les dirigeants qui adoptent des méthodes administratives de management des activités techniques en tirent effectivement des avantages, ceux que j'ai décrits ci-dessus, notamment en termes de renforcement du pouvoir administratif et de diminution de l'exigence de compétence.
- Jean-Pierre Dupuy a emprunté à Friedrich von Hayek une théorie qui est de plus en plus utilisée par les économistes, et qui étudie les phénomènes d'imitation au sein de l'économie de marché. Alors que l'économie néo-classique se représente un *homo œconomicus* autosuffisant et indépendant, parfaitement informé et rationnel dans des choix censés le mener à un optimum qui, à l'échelle du marché, produirait un équilibre, Hayek met en évidence,

> après Adam Smith et Keynes, le rôle central de l'*imitation* dans les phénomènes collectifs dont le marché est le cadre. Le rôle de l'imitation semble particulièrement important dans les situations de choix entre techniques rivales, et aucun mécanisme ne garantit que la technique qui va l'emporter sera la meilleure. En effet, dans le jeu de miroirs qui précède l'engouement mimétique, une simple rumeur peut orienter quelques acteurs vers la cible, ce qui déclenchera un effet d'avalanche : « [l'imitation généralisée] suscite des dynamiques auto-renforçantes qui convergent si résolument vers leur cible qu'il est difficile de croire que cette convergence n'est pas la manifestation d'une nécessité sous-jacente... ». Nous ne saurions écarter l'hypothèse que le succès universel des méthodes de gestion de projet pourrait résulter d'un phénomène mimétique de ce type : dit en d'autres termes, pour citer un proverbe du réseau, « 100 000 lemmings ne peuvent pas avoir tort ».

De ce qui précède peut-on déduire qu'il faut forcément être ingénieur informaticien pour devenir directeur du système d'information ? Non, mais un DSI (et d'ailleurs tout dirigeant) devra posséder, pour remplir ses fonctions, un certain nombre de compétences, et il ne pourra pas faire face aux problèmes qui se posent à lui uniquement avec des procédures administratives normalisées. Le rôle de l'informatique dans le monde contemporain est tel que nul ne peut plus se passer d'en connaître les techniques de base.

Dans le contexte français où l'absence de compétence technique est devenue un atout déterminant pour l'accès aux postes de direction des systèmes d'information, les méthodes de management de système selon les normes IS 9001 et IS 27001 acquièrent la propriété de prédictions autoréalisatrices : pour les raisons évoquées ci-dessus de nombreux DSI ont d'ores et déjà emprunté cette démarche, et leurs collègues en retard, qui n'ont pour boussole dans cet univers que l'air du temps et le qu'en dira-t-on, trouveront facilement auprès de leurs pairs la confirmation que c'est bien dans cette voie qu'il faut aller. Les sommes considérables englouties par ces méthodes n'apparaissent pas forcément comme des inconvénients, puisqu'elles renforcent l'importance et le prestige de celui qui les ordonne, et donnent satisfaction à la direction générale, qui ne dispose en général ni des informations ni des moyens d'investigation nécessaires pour se former une opinion sur le sujet, et qui peut faire état du recours à ces méthodes éprouvées pour répondre aux questions des auditeurs ou des actionnaires.

Quant à nous, nous nous efforcerons au cours des chapitres suivants de dispenser les principes de sécurité substantielle qui nous semblent le socle de ce que doit être aujourd'hui un système sûr, et que plus grand monde ne peut se permettre d'ignorer totalement, que ce soit dans l'entreprise ou dans l'usage privé des ordinateurs et des réseaux.

Richard Feynman à propos de la conduite de projet

Un des derniers écrits du physicien Richard P. Feynman, prix Nobel 1965, fut une annexe [50] au rapport de la Commission Rogers rédigé à la demande des autorités gouvernementales américaines à la suite de l'accident dramatique de la navette spatiale Challenger et destiné à en élucider les circonstances. Il y a suffisamment de points communs entre un sinistre spatial et un sinistre informatique pour que les leçons tirées de celui-là puissent être utiles à ceux qui se préoccupent de celui-ci ; en effet, si les objets produits par l'industrie spatiale et par l'industrie informatique paraissent très dissemblables, les méthodes de conduite de projet mises en œuvre dans l'un et l'autre cas puisent à la même source d'inspiration (le projet Apollo dans les années 1960), et risquent donc d'avoir des effets similaires. En outre, même si le risque semble bien moindre de mettre en danger des vies humaines dans le second cas que dans le premier, il convient de noter qu'une navette spatiale incorpore des millions de lignes de logiciel informatique, soit embarqué soit dans les installations au sol, sans oublier les programmes qui ont servi à sa conception. Il n'y a donc aucune raison de se priver des enseignements prodigués à cette occasion par un des scientifiques du XXe siècle les plus réputés, notamment pour ses talents pédagogiques.

Pour établir son rapport, R. Feynman a rencontré différents experts qui avaient participé à la conception et à la réalisation de la navette spatiale, ou qui avaient donné des consultations à son sujet avant ou après l'accident, et il a lu leurs rapports. Il a été frappé par la discordance extraordinaire, parmi les experts et les officiels de la NASA, des opinions relatives au risque d'accident mortel, puisqu'elles vont de 1 accident sur 100 vols à 1 accident sur 100 000 vols, où les premières émanent surtout des ingénieurs qui ont réellement travaillé sur le projet, et les dernières plutôt des managers. Il a également observé la diminution au fil du temps de la sévérité des critères de certification, au fur et à mesure que les vols sans incidents instauraient l'idée que « puisque le risque avait été encouru jusqu'à

présent sans qu'un accident survienne, il pouvait être accepté pour la prochaine fois ».

Pour ce qui nous concerne ici, la passage le plus intéressant du texte est celui qui a trait aux moteurs à combustible liquide de la navette *(Space Shuttle Main Engines, SSME)*. Ces composants sont parmi les plus complexes de l'ensemble. Feynman explique que la méthode habituelle de conception de tels moteurs (par exemple pour des avions civils ou militaires) procède selon une démarche *de bas en haut (bottom up)* : on commence par étudier les caractéristiques souhaitables des matériaux à utiliser, puis on teste des pièces élémentaires au banc d'essai. Sur la base des connaissances acquises ainsi, on commence à tester des sous-ensembles plus complexes. Les défauts et les erreurs de conception sont corrigés au fur et à mesure : comme ils ne portent que sur des parties de l'ensemble, les coûts sont modérés. Si des défauts sont encore détectés au moment de l'assemblage de l'ensemble, ils restent relativement faciles à localiser et à corriger, notamment du fait de l'expérience acquise par les tests de sous-ensembles.

Or les moteurs à combustible liquide de la navette n'ont pas été conçus selon cette démarche *bottom up*, mais selon l'approche inverse, de *haut en bas (top down)*, c'est-à-dire que le moteur a été conçu et réalisé tout en même temps, avec très peu d'études et d'essais préalables des matériaux et des composants ; avec une telle démarche, la recherche de l'origine d'un défaut ou d'une erreur de conception est beaucoup plus difficile qu'avec la méthode *bottom up*, parce que l'on dispose de peu d'informations sur les caractéristiques des composants. Il faut alors utiliser le moteur complet comme banc d'essai pour trouver la panne, ce qui est très difficile et onéreux. Il est en outre difficile dans ces conditions d'acquérir une compréhension détaillée des caractéristiques et du fonctionnement du moteur, compréhension qui aurait été de nature à fonder la confiance que l'on aurait pu avoir en lui.

La méthode *top down* a un autre inconvénient : si l'on trouve une erreur de conception sur un sous-ensemble, comme la conception n'en a pas été isolée, mais intégrée dans la conception d'ensemble, il faut repenser la conception générale. Il est à craindre que pour des erreurs jugées mineures (à tort ou à raison), la lourdeur des investigations à entreprendre n'incite à renoncer à reprendre la conception de l'ensemble, alors qu'il faudrait le faire.

Nous pensons que cette critique de la méthode *top down* par Richard P. Feynman s'applique bien aux systèmes informatiques, et particulièrement aux systèmes de sécurité informatique. Mais ne lui faisons pas dire ce qu'elle ne dit pas : il convient bien sûr d'avoir une vision d'ensemble du système, simplement il ne faut pas lui accorder les vertus qu'elle n'a pas, elle ne doit pas être trop précise, ce n'est pas d'elle qu'il faudra déduire la conception détaillée des éléments et des sous-systèmes.

2

Les différents volets de la protection du SI

Avec ce chapitre nous entamons la teneur technique de notre sujet, mais en douceur : les droits d'accès et leur vérification, l'authentification et le chiffrement sont décrits en termes généraux... ainsi que certaines attaques dont ils peuvent faire l'objet.

L'indispensable sécurité physique

Avant d'entrer plus avant dans le vif de notre propos, il convient de faire un détour par un sujet que nous ne traiterons pas en détail, mais qu'il importe d'évoquer : toute mesure de protection logique est vaine si la sécurité physique des données et des traitements n'est pas convenablement assurée. Il convient donc d'accorder un soin jaloux aux points suivants :

- qualité du bâtiment qui abrite données et traitements, à l'épreuve des intempéries et des inondations, protégé contre les incendies et les intrusions ;

- contrôles d'accès adéquats ;
- qualité de l'alimentation électrique ;
- certification adéquate du câblage du réseau local et des accès aux réseaux extérieurs ; la capacité des infrastructures de communication est très sensible à la qualité physique du câblage et des connexions ;
- pour l'utilisation de réseaux sans fil, placement méticuleux des bornes d'accès, réglage de leur puissance d'émission et contrôle des signaux en provenance et à destination de l'extérieur.

Ces précautions prises, il faut néanmoins envisager qu'elles puissent se révéler insuffisantes, et que l'intégrité physique de votre système d'information soit alors compromise. La *compromission* d'un système d'information désigne le fait qu'un intrus ait pu, d'une façon ou d'une autre, en usurper l'accès pour obtenir des informations qui auraient dû rester confidentielles. Pour éviter que cette circonstance n'entraîne la disparition de l'entreprise, il aura fallu prendre les mesures suivantes :

- sauvegarde régulière des données sur des supports physiques adéquats distincts des supports utilisés en production ;
- transport régulier de copies de sauvegarde en dehors du site d'exploitation ;
- aménagement d'un site de secours pour les applications vitales.

Ces précautions seront inopérantes si elles ne font pas l'objet d'une documentation tenue à jour et d'exercices périodiques : en situation de catastrophe, il s'avère que les humains ne savent faire que ce à quoi ils sont entraînés, des actions complexes qui n'auront jamais été effectuées « à blanc » ne pourront avoir pour conséquence qu'une catastrophe encore plus grave.

Des solutions techniques existent pour toutes ces mesures, mais leur mise en œuvre est complexe et onéreuse, ce qui conduit souvent à les négliger. Le débit des réseaux modernes permet de disposer à plusieurs kilomètres du site d'exploitation un site miroir dont les données pourront être mises à jour heure par heure, ou même en temps réel si cela est vraiment indispensable, le coût n'est même pas tellement élevé, mais la conception et la réalisation d'une telle organisation sont loin d'être des tâches faciles. De même, la complexité d'un plan de sauvegarde pour quelques dizaines de serveurs en réseau ne doit en aucun cas être sous-estimée.

La sécurité des données peut également être améliorée par le recours aux possibilités des matériels modernes de stockage et de leurs logiciels de pilotage : les systèmes NAS *(Network Attached Storage)* offrent des possibilités intéressantes de

prise d'instantanés (snapshots) et de réplication à distance, les batteries de disques RAID et les systèmes de fichiers virtuels tels que *Logical Volume Management* (LVM) diminuent grandement les risques de perte de données en cas de défaillance d'un disque. Le lecteur sera bien avisé de s'intéresser à ces sujets, qui font l'objet de nombreux et volumineux ouvrages, et dont nous ne saurions donner ici plus que cette énumération brève et non exhaustive ; on pourra se reporter à mon propre texte[1] pour une introduction succincte, et à l'excellente synthèse de Curtis Preston [86] pour une explication complète et une bibliographie.

Les mesures évoquées ici sont des missions pour des ingénieurs spécialisés, de haut niveau, et surtout expérimentés. Nous n'entrerons pas plus dans les détails de ces actions, mais nous ne saurions trop mettre en garde contre la tentation de les négliger.

Protéger le principal : le système d'exploitation

Afin d'être fiable, un système d'exploitation digne de ce nom doit comporter des dispositifs et des procédures de protection des objets qu'il permet de créer et de manipuler. Les objets à protéger appartiennent à deux grandes catégories : les objets persistants tels que les fichiers, et les objets éphémères créés en mémoire pendant l'exécution d'un processus et destinés à disparaître avec lui. Les objets matériels, tels que périphériques physiques, interfaces réseau, etc., sont assimilés à des objets persistants. La protection consiste à empêcher qu'un utilisateur puisse altérer un fichier qui ne lui appartient pas sans que le propriétaire lui en ait donné l'autorisation, ou encore, par exemple, à empêcher qu'un processus en cours d'exécution ne modifie une zone mémoire attribuée à un autre processus sans l'autorisation du propriétaire de celui-ci.

Droits d'accès

De façon très générale, la question de la protection d'un objet informatique se pose dans les termes suivants, inspirés des concepts mis en œuvre par le système Multics [80] (voir aussi CROCUS [37]) :

[1] `http://www.laurent-bloch.org/Livre-Systeme/livre008.html`.

- Un objet a un propriétaire identifié, généralement l'utilisateur qui l'a créé. Un objet est, sous réserve d'inventaire, soit un fichier, soit un processus, soit des structures de données éphémères créées en mémoire par un processus, mais pour Multics tous ces objets sont en fin de compte des espaces de mémoire virtuelle nommés segments ou sont contenus dans des segments.
- Le propriétaire d'un objet peut avoir conféré à lui-même et à d'autres utilisateurs des droits d'accès à cet objet. Les types de droits possibles sont en général les suivants (on peut en imaginer d'autres) :
  - droit d'accès en consultation (lecture) ;
  - droit d'accès en modification (écriture, destruction, création) ;
  - droit d'accès en exécution ; pour un programme exécutable, la signification de ce droit est évidente ; pour un répertoire de fichiers ce droit confère à ceux qui le possèdent la faculté d'exécuter une commande ou un programme qui consulte ce répertoire ;
  - droit de blocage, par exemple pour un processus en cours d'exécution ou éligible pour l'exécution.
- À chaque objet est donc associée une liste de contrôle d'accès *(access control list)* qui énumère les utilisateurs autorisés et leurs droits.
- Avant toute tentative d'accès à un objet par un utilisateur, l'identité de cet utilisateur doit être authentifiée.
- Pour qu'un utilisateur ait le droit d'exécuter une action sur un objet, et dans un système informatique cette action est perpétrée par l'entremise d'un processus, il faut en outre que le processus en question possède le *pouvoir* voulu. Le pouvoir est un attribut d'un processus, il peut prendre des valeurs qui confèrent à ce processus des *privilèges* plus ou moins étendus. La plupart des systèmes ne proposent que deux valeurs de pouvoir : le mode superviseur, qui confère le pouvoir absolu, et le mode utilisateur, qui limite les actions de l'utilisateur en question aux objets dont il est propriétaire. Mais nous allons voir que certains systèmes ont affiné la hiérarchie des valeurs de pouvoir.
- La valeur du pouvoir d'un processus peut changer au cours de son exécution. Ainsi un processus qui se déroule dans un mode utilisateur peut faire une demande d'entrée-sortie, ce qui nécessite le mode superviseur. Ceci sera résolu, sous Unix par exemple, par le mécanisme de l'appel système, qui trans-

fère le contrôle, pour le compte du processus utilisateur, à une procédure du noyau qui va travailler en mode superviseur.

- Nous définirons la notion de *domaine de protection* dans lequel s'exécute un processus comme l'ensemble des objets auxquels ce processus a accès et des opérations qu'il a le droit d'effectuer sur ces objets. Lorsqu'un processus change de valeur de pouvoir, il change par là-même de domaine de protection.

Vérification des droits, imposition des protections

Les dispositifs et procédures de protection du système d'exploitation vont consister à faire respecter les règles qui découlent des droits et pouvoirs énumérés ci-dessus et à empêcher leur violation. La protection au sens où nous allons l'étudier dans ce chapitre ne consiste pas à empêcher les erreurs humaines, les défaillances techniques ou les actes de malveillance qui pourraient faire subir à un objet un sort non désiré, mais seulement à empêcher leur incidence sur les objets en question. Il faut protéger les données et les processus d'un utilisateur contre les processus des autres utilisateurs, protéger le fonctionnement du système contre les processus des utilisateurs et vice-versa, enfin protéger les uns des autres les processus d'un même utilisateur.

La qualité des dispositifs et procédures de protection fait la *sûreté* d'un système d'exploitation. On conçoit notamment aisément que le contrôle des droits et des pouvoirs doive être à l'abri des manipulations d'utilisateurs désireux sans légitimité d'accroître leurs privilèges, ce qui signifie que les procédures de contrôle doivent s'exécuter avec le mode de pouvoir le plus grand et les droits les plus étendus, inaccessibles aux utilisateurs ordinaires. Cette réflexion de bon sens suffit à refuser le qualificatif « sûr » à tel système d'exploitation qui comporte un système perfectionné de listes d'accès réalisé... en mode utilisateur, et pour lequel de surcroît l'identification des utilisateurs est facultative.

En effet, et cela va sans dire, mais disons-le : il ne sert à rien de contrôler les droits et les pouvoirs du propriétaire d'un processus si son identité n'est pas déjà raisonnablement certaine. Les procédures d'identification et d'authentification des utilisateurs sont un préalable à toute stratégie de protection.

Gérer l'authentification

Les sections précédentes ont présenté les principes de conception des dispositifs de protection fournis par les systèmes d'exploitation modernes : il convient de garder à l'esprit que ces dispositifs ne seront efficaces que s'ils sont effectivement utilisés selon des *politiques de sécurité* dont les grandes lignes feront l'objet de la suite de ce chapitre.

Séparation des privilèges

La *séparation des privilèges* consiste à attribuer à chaque utilisateur, ou à chaque activité du système, les privilèges dont il a besoin, et pas d'autres. C'est le principe du *privilège minimum* qui doit s'appliquer ici.

Ainsi, sur un système Unix par exemple, il existe un utilisateur `root` doté de tous les droits sur tous les objets du système : il peut créer, détruire ou modifier tous les fichiers, lancer ou interrompre toutes les activités et tous les programmes. Autant dire qu'il est très dangereux de commettre une fausse manœuvre lorsque l'on est connecté au système sous le compte `root`, puisque l'on peut par exemple effacer l'ensemble d'un système de fichiers, ou corrompre des fichiers de paramètres du système de telle sorte qu'il ne pourra plus fonctionner, ou que sa sécurité sera fortement compromise. Le principe de séparation des privilèges commande de n'utiliser le compte `root` que le moins souvent possible, de préférence jamais. Il existe des dispositifs qui permettent aux administrateurs du système d'accroître leur niveau de privilèges autant que de besoin, et quand il est besoin, pour une action précise, et pour celle-ci seulement. De surcroît, les journaux du système, destinés à garder la trace de tous les événements significatifs qui y surviennent, enregistreront l'identité réelle des auteurs des actions qui ont nécessité une augmentation de privilège, ce qui est également indispensable à une bonne administration de la sécurité.

De même, certains programmes sont destinés à exécuter des actions privilégiées, mais cantonnées à une partie délimitée du système. Le serveur central d'un système de bases de données doit disposer des privilèges qui lui donneront les moyens de commettre toutes actions nécessaires sur les bases de données, mais uniquement dans ce domaine. De même pour le logiciel de sauvegarde ou pour le serveur de transfert du courrier électronique : ces programmes devront être exécutés sous le compte d'un pseudo-utilisateur spécial, doté uniquement des privilèges néces-

saires pour son domaine d'action, et pour lui seul. De tels serveurs ne doivent pas être lancés sous le compte de l'utilisateur `root`. Les outils de configuration des systèmes Unix modernes, tels que Linux ou OpenBSD, établissent automatiquement ces mesures de séparation des privilèges, et il convient de n'y rien modifier.

Quant aux utilisateurs ordinaires, il convient de ne leur conférer de droits d'accès en création, en écriture et en destruction qu'à leurs propres données, et en lecture aux données partagées, autant que de besoin.

Nous évoquerons à nouveau, de façon plus technique, la question de la séparation des privilèges à la page 101.

Identification et authentification

Dès lors que l'identité d'un utilisateur du système détermine ses privilèges et ses droits d'accès à telles ou telles données, il convient que cette identité soit correctement administrée, qu'elle ne puisse pas être usurpée par un tiers, et que son authenticité puisse être vérifiée.

Avant toute chose, les utilisateurs doivent être convaincus du caractère privé de leur identité : cela semble évident, mais de mauvaises habitudes héritées des premiers temps de l'informatique conduisent encore beaucoup de systèmes à être utilisés par plusieurs personnes sous un compte unique dont tout le monde connaît le mot de passe : une telle habitude doit être combattue sans relâche, parce que sur un tel système aucune sécurité n'existe ni ne peut exister. L'interdiction de telles pratiques devrait figurer dans une charte d'usage des systèmes et des réseaux, validée par les instances de concertation telles que le comité d'entreprise et annexée au règlement intérieur.

Pour qu'un système d'identification soit efficace, il faut que l'utilisateur puisse se l'approprier facilement. On évitera donc autant que possible d'avoir un système d'identification particulier pour chaque application, et on se dirigera plutôt vers les systèmes d'identification centralisés, par exemple le *Single Sign-On (SSO)* ; l'encadré ci-après précise ces notions.

Le *Single Sign-On* (SSO)

Le souhait légitime de tout utilisateur d'un système informatique est la simplicité d'utilisation. En de nombreux endroits il y a encore, pour identifier une seule et même personne une multiplicité d'identifiants et de mots de passe, cela pourrait être :

- mon `login` de messagerie est `jdupont` ;
- mon `login` pour accéder à l'application de gestion des congés payés `jdu`, etc.

Certains organismes ont déjà fait des efforts pour faciliter la vie de leurs utilisateurs : un seul et même *login* est utilisé pour la totalité des applications. Toutefois l'utilisateur doit ressaisir celui-ci et le mot de passe associé chaque fois qu'il ouvre son application. Il n'est pas rare de devoir saisir ces données plusieurs dizaines de fois en une journée de travail. En plus de l'agacement lié à la saisie multiple de l'identifiant et du mot de passe il y a des risques de sécurité réels :

- ces données sont probablement dupliquées dans les formats natifs de chacune des applications ;
- puis-je faire confiance au format de stockage des mots de passe des applications ?
- la gestion du changement de mot de passe devient plus complexe.

Les objectifs des solutions dites de *SSO* sont donc multiples :

- avoir un référentiel centralisé des identités et des mots de passe, référentiel auquel les applications s'adressent pour vérifier l'identité d'un utilisateur ;
- ne jamais divulguer le mot de passe (même sous une forme chiffrée) à une application, au pire l'application fournira au système de *SSO* les identifiants reçus de l'utilisateur pour validation et au mieux un système de tickets (inspiré de Kerberos) entre les trois acteurs que sont l'utilisateur (et son navigateur Web), le serveur d'authentification du *SSO* et l'application, ce qui permet de ne jamais transmettre les éléments secrets (le mot de passe ou ce qui le remplace) à l'application ;
- éviter les saisies multiples de l'identité pour accéder à plusieurs applications.

C'est ce dernier point qui est bien sûr le plus important aux yeux de l'utilisateur final, mais c'est aussi le plus difficile à mettre en œuvre. Si la prise en charge de tels systèmes prend une certaine ampleur avec des applications « webisées », grâce à l'utilisation de HTTPS (HTTP (pour *Hypertext Transport Protocol*) est le protocole de circulation des données sur le Web ; HTTPS en est la version sécurisée par chiffrement) et de *cookies* (les *cookies* sont des éléments de données qui permettent de conserver l'identification d'un utilisateur sur le Web pour une série de transactions), la mise en œuvre d'un tel système dans le cadre d'autres applications dites « lourdes » est, elle, plus complexe même impossible car elle ne permet pas d'utiliser les mêmes flux de données entre l'utilisateur final, l'application et le serveur d'authentification. Le lecteur désireux d'approfondir sa connaissance du fonctionnement, complexe, d'un système de *SSO* pourra utilement lire deux présentations faites aux JRES 2003 par Olivier Salaün (Olivier Salaün. « Introduction aux architectures Web de *Single-Sign On* ». 2003.

http://2003.jres.org/actes/paper.116.pdf) et par Pascal Aubry et ses collègues (Pascal Aubry, Julien Marchal, et Vincent Mathieu. « *Single Sign-On Open Source* avec CAS ». 2003. http://2003.jres.org/actes/paper.139.pdf).

Mettre en œuvre et exploiter un tel système apporte un confort indéniable, mais cela a un coût : l'investissement éventuel pour une solution du marché, les compétences requises pour la mise en œuvre et l'exploitation de tous les jours, et enfin, l'adaptation des applications « Web » au produit retenu. De l'avis même de ceux qui ont déployé de telles solutions, le plus difficile est l'adaptation des applications au système de *SSO* retenu.

Le bon vieux mot de passe

En cette année 2006, le procédé d'authentification le plus utilisé est sûrement encore, de loin, et malgré ses faiblesses bien connues, le couple identifiant–mot de passe *(login–password)*. Rappelons-en brièvement le principe, en prenant l'exemple d'un système Unix (ou Linux, qui, rappelons-le, n'est qu'une variété d'Unix).

Sous Unix, la création du compte d'un utilisateur crée une entrée dans le fichier `/etc/passwd`. Ce fichier contient une entrée par utilisateur, chaque entrée comporte plusieurs champs séparés les uns des autres par le caractère `:` ; les champs sont :

- le *nom d'utilisateur*, ou identifiant, qui est une chaîne de caractères qui identifie de façon unique cet utilisateur ; cet identifiant est aussi appelé *nom de login* ;
- la représentation chiffrée du *mot de passe* de l'utilisateur ; nous verrons d'ici peu qu'en fait cette représentation chiffrée est conservée ailleurs que dans le fichier `/etc/passwd`, pour des raisons de sécurité ;
- le numéro d'identification (`uid`) de l'utilisateur ;
- le numéro de groupe (`gid`) de l'utilisateur ;
- le *vrai nom* de l'utilisateur ;
- son répertoire d'accueil ;
- son programme d'accueil *(shell)*.

Le fonctionnement d'Unix exige que le fichier `/etc/passwd` soit accessible en lecture par tout le monde : cela ouvrait à des malveillants la possibilité de récupérer les mots de passe chiffrés des utilisateurs, puis d'essayer de les « casser »

tranquillement sur leur ordinateur. Sachant que la fonction qui prend un mot de passe « en clair » pour le chiffrer est publique, les méthodes les plus ordinaires pour essayer de casser un mot de passe sont :

- l'attaque par force brute, qui consiste à essayer successivement toutes les combinaisons possibles de caractères pour générer des mots de passe arbitraires, les chiffrer et comparer le résultat au texte chiffré obtenu par le « pompage » de `/etc/passwd` ;
- l'attaque par dictionnaire, où l'on utilise une liste de mots courants et où l'on essaye successivement toutes les variations orthographiques ou typographiques possibles de ces mots, chiffrés au moyen de l'algorithme utilisé, pour ici aussi comparer le résultat au texte chiffré de `/etc/passwd`.

Ces méthodes, surtout la seconde, donnent généralement des résultats positifs, alors pour éviter ce risque les mots de passe chiffrés ne sont en général plus conservés dans le fichier `/etc/passwd`, mais dans un fichier aux droits d'accès plus restreints, `/etc/shadow`.

Listes de contrôle d'accès

Comme nous l'avons signalé ci-dessus page 38, les listes de contrôle d'accès *(access control list, ACL)* procurent une séparation des droits et privilèges plus fine que les dispositifs standard d'un système comme Unix, en permettant d'accorder des autorisations à un utilisateur particulier pour un fichier particulier.

Les liste de contrôle d'accès pour le réseau seront décrites dans le chapitre 6, à la page 132

ACL Posix

Une ACL POSIX est associée à un fichier ou à un répertoire ; elle est constituée d'une liste d'entrées qui appartiennent à l'un des types suivants :

Type d'entrée	Forme de l'entrée
Propriétaire	`user : :`*`rwx`*
Utilisateur nommé	`user :`*`nom :rwx`*
Groupe propriétaire	`group : :`*`rwx`*
Groupe nommé	`group :`*`nom :rwx`*
Masque	`mask : :`*`rwx`*
Autres	`other : :`*`rwx`*

Les lettres `rwx` signifient *read, write, execute* (lire, écrire, exécuter) et désignent les trois types d'autorisation que peut désigner une entrée d'ACL. Ainsi, la liste suivante :

```
user::rw-
user:pierrot:rw-
group::r--
mask::rw-
other::---
```

donne-t-elle, pour le fichier concerné, les droits de lecture et d'écriture au propriétaire du fichier et à l'utilisateur Pierrot, les droits de lecture au groupe du propriétaire, et aucun droit aux autres utilisateurs.

Utiliser les ACL Posix sur un système de fichiers d'un ordinateur sous Linux nécessite une modification de ce système de fichiers afin d'y introduire les données supplémentaires propres aux ACL. Pour les systèmes *Windows* récents les ACL sont disponibles de façon standard.

Historique des ACL

Les ACL sont apparues en 1984 dans la version 4 du système d'exploitation VMS *(Virtual Memory System)* pour les ordinateurs VAX de *Digital Equipment*. L'idée en a été reprise par *Cisco* pour le système IOS *(Internet Operating System)* de ses routeurs. Les ACL font depuis 2002 l'objet d'une norme POSIX, mais leur implantation dans les différents systèmes d'exploitation reste assez hétérogène.

Le chiffrement asymétrique

Nous allons donner ici une première présentation (une seconde présentation, plus technique, prendra place au chapitre 4 page 71) d'une famille de méthodes d'authentification, de signature et de chiffrement : le *chiffrement asymétrique*, famille à laquelle appartiennent notamment les méthodes dites *à clé publique*. Ces méthodes sont aujourd'hui largement utilisées ; nous verrons que l'authentification et la signature sont en fait des usages particuliers du chiffrement, ce pourquoi cette section sort de son cadre initial, qui était l'authentification, pour s'aventurer vers le chiffrement, qui pourrait sembler un sujet assez éloigné.

Pour reprendre les termes de Christian Queinnec dans la préface du présent ouvrage, l'invention par Withfield Diffie et Martin E. Hellman [40] d'une nou-

velle méthode d'échange de clés a permis de résoudre un problème ouvert depuis des millénaires : comment deux personnes, ne se connaissant au préalable pas, peuvent-elles élaborer un secret commun à elles seules en n'ayant échangé que des messages publics ? Cette révolution ouvrait la voie à l'invention par Ronald Rivest, Adi Shamir et Leonard Adleman du *chiffrement asymétrique* [91], qui permet à tout un chacun de publier sur son site Web la clé de chiffrement à utiliser pour lui envoyer un message secret.

Les principes mathématiques du chiffrement asymétrique sont exposés (de façon aussi simple que possible) au chapitre 4 page 71. Nous allons ici expliquer comment il est utilisé et pourquoi il est si intéressant, en le considérant (provisoirement donc) comme une boîte noire, ou comme une fonction mathématique nommée *Chiffrer*, dont la fonction inverse serait $\mathit{Chiffrer}^{-1}$.

Ainsi, si $x \xrightarrow{\mathit{Chiffrer}} y$, alors $y \xrightarrow{\mathit{Chiffrer}^{-1}} x$.

L'idée de base du chiffrement asymétrique, c'est que la fonction *Chiffrer* est *difficilement inversible* : calculer *Chiffrer* est (relativement) facile, mais calculer $\mathit{Chiffrer}^{-1}$ est très difficile, en pratique impossible.

Nous souhaitons pouvoir faire quatre choses avec notre système de chiffrement :

signer un document	vérifier l'authenticité d'une signature
chiffrer un document à envoyer	déchiffrer un document reçu

Nous pourrions ajouter : s'authentifier lors de l'accès à un service, et vérifier une tentative d'authentification. Ce sont des cas particuliers de signature.

Les impératifs à respecter sont les suivants :

- la signature électronique doit être difficile à falsifier (en pratique, ce doit être impossible, mais cette impossibilité n'est pas démontrable mathématiquement) ;
- en d'autres termes, il doit être difficile de se procurer les dispositifs qui peuvent fabriquer la signature, pour fabriquer par exemple un faux ;
- l'authenticité d'une signature doit en revanche être facile à vérifier ;
- de même, il doit être difficile de déchiffrer un document chiffré quand on n'en est pas le destinataire légitime, c'est-à-dire en « cassant » le code ;
- en revanche il doit être facile de chiffrer un document de sorte que seul son destinataire légitime soit en mesure de le déchiffrer.

Sans anticiper sur le chapitre 4 consacré à une étude plus détaillée des méthodes de chiffrement, et en résumant à l'extrême, on peut énoncer la chose ainsi :

- la clé secrète $Ksec$, qui permettra de signer un document de manière infalsifiable ou de déchiffrer un document chiffré, sera un couple de grands nombres premiers : p, q, qui auront chacun de l'ordre de 150 chiffres décimaux ;
- la clé publique $Kpub$, publiée dans un annuaire ou sur un site Web, sera le produit de ces deux nombres : $p \times q$.

En disant cela on glisse sur quelques détails techniques, mais le fond de la question est celui-là. L'idée est la suivante : il est très facile, connaissant p et q, de calculer $p \times q$, mais très difficile, connaissant $p \times q$, d'en déduire p et q si ces deux nombres sont suffisamment grands.

Chiffrement et déchiffrement

Si Aïcha veut envoyer un message secret M à Berthold (ces prénoms sont choisis parce que l'auteur est las des sempiternels Alice et Bob), elle récupère la clé publique de Berthold (on chiffre *toujours* avec la clé publique du destinataire) $Kpub$ dans l'annuaire de son site Web, par exemple, et le chiffrement consiste en une transformation mathématique simple pour obtenir le chiffré C :

$$C = \textit{Chiffrer}(M, Kpub_{\text{du destinataire}})$$

Pour déchiffrer, Berthold utilise sa clé privée :

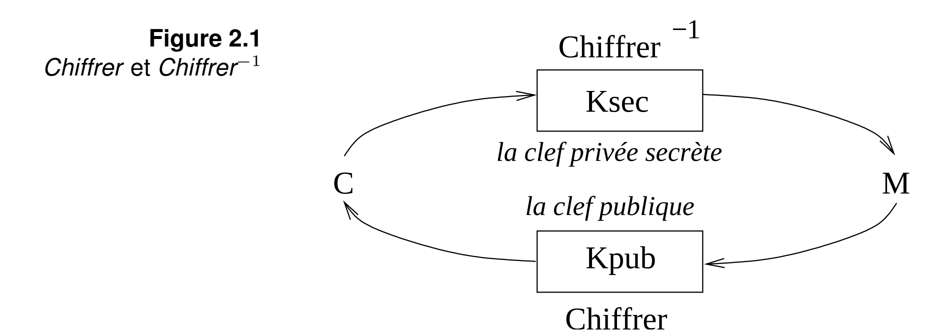
$$M = \textit{Chiffrer}^{-1}(C, Ksec_{\text{du destinataire}})$$

Signature et vérification

Supposons maintenant que le message d'Aïcha ne soit plus secret, mais qu'elle veuille le signer de telle sorte que Berthold soit certain de son origine, donc qu'il émane bien d'elle, et que ce ne soit pas par exemple un faux fabriqué par le méchant Jean-Kevin.

Outre sa fonction de chiffrement, *Chiffrer* est aussi utilisable de façon très simple pour signer de façon sûre et non répudiable un document. Il est important qu'une

Figure 2.1
Chiffrer et *Chiffrer*$^{-1}$



signature ne puisse pas être répudiée, c'est-à-dire que le signataire ne puisse pas prétendre qu'il n'est pas l'auteur du document signé, que cette signature n'est pas son œuvre. La qualité de résistance à la répudiation doit résider dans une preuve de la signature, détenue par le destinataire du document signé, vérifiable par un tiers, et inaltérable par le signataire. Pour conférer cette qualité prisée à sa signature, il suffit que le signataire la chiffre avec sa clé *privée* : le destinataire la déchiffrera avec la clé publique du signataire, et si le déchiffrement réussit ce sera la preuve que la signature est authentique, en d'autres termes une authentification sûre.

Une autre méthode consiste à signer un « résumé numérique » du message. Ce résumé, appelé *condensat*, est produit par un algorithme de condensation, tel MD5 créé par Ronald Rivest, ou SHA (*Secure Hash Standard* FIPS 180-1). Le principe d'une fonction de condensation (parfois appelée *hachage*) est le suivant : soient M et M' deux messages, et H la fonction :

1. si $M \neq M'$, la probabilité que $H(M) = H(M')$ est très voisine de 0 ;
2. quel que soit M, il est difficile de trouver un $M' \neq M$ tel que $H(M') = H(M)$.

Cette propriété d'un algorithme de condensation que l'on ne puisse pas trouver facilement deux textes différents qui donnent le même résumé est appelée la *résistance aux collisions*, elle est essentielle.

Un auteur peut par conséquent signer en calculant le condensat de son message, en le chiffrant grâce à sa clé privée puis en le diffusant. Tout détenteur de sa clé publique et du message sera en mesure de vérifier la signature.

Outre une signature non répudiable, ce procédé garantit en pratique l'intégrité du message.

La signature est effectuée en deux temps :

- Le logiciel (de courrier électronique, par exemple) de l'ordinateur d'Aïcha calcule un résumé du message selon un des algorithmes convenus et publics, tels que MD5 ou SHA-1, qui répondent aux conditions suivantes :
  - connaissant le résumé R d'un message M, il est très difficile de fabriquer un message M' différent auquel corresponde le même résumé R ;
  - la probabilité que deux messages donnent le même résumé est très faible.
- Aïcha chiffre le résumé avec sa clé *privée*.

Lorsque Berthold reçoit le message d'Aïcha, il fait deux choses (ou plutôt c'est son logiciel de courrier électronique qui le fait) :

- il déchiffre le résumé chiffré avec la clé *publique* d'Aïcha ;
- il calcule le résumé du message par le même algorithme qu'Aïcha ;
- si les deux résumés sont égaux, le message a bien été signé par Aïcha, seule détentrice de sa clé privée.

Dans le cas d'un message chiffré ou signé envoyé à plusieurs destinataires, les méthodes utilisées en pratique emploient des algorithmes propres à éviter de reproduire le corps du message en autant d'exemplaires que de destinataires ; ces méthodes sont décrites à la page 174.

Comprendre les failles et les attaques sur les logiciels

L'idée du chiffrement asymétrique avec un couple clé publique-clé privée semble tellement puissante qu'on ne voit pas de raison pour qu'elle ne supplante pas toutes les autres techniques. En fait un algorithme aussi puissant soit-il ne résout pas tous les problèmes. D'abord les algorithmes de chiffrement asymétriques tel RSA sont lourds en temps de calcul, ce qui peut s'avérer dissuasif, mais les cryptosystèmes à clés publiques sont exposés à d'autres menaces, dont l'examen nous confirmera qu'il n'existe pas de solution purement technique aux questions de sécurité.

L'attaque par interposition (Man in the middle)

Le meilleur chiffrement du monde ne peut pas empêcher qu'un agent mal intentionné, disons Charles, se soit fait passer pour Franz, ait intercepté les communications d'Anne, et lui ait présenté sa clé publique comme étant celle de Franz : ainsi Charles pourra facilement déchiffrer les messages d'Anne avec sa propre clé privée, les lire, puis les re-chiffrer avec la vraie clé publique de Franz et les faire parvenir à ce dernier. Ce type d'attaque, appelé *Man in the middle* (par interposition), est difficile à déjouer une fois que Charles a réussi à s'introduire dans le circuit de communication ; elle peut être tentée contre RSA et aussi contre l'algorithme de Diffie-Hellman, qui sera décrit à la page 75 et sur lequel reposent souvent les procédures d'échange de clés.

Pour éviter le coût en termes de temps de calcul du chiffrement asymétrique, tout en bénéficiant de la sécurité qu'il procure, certains protocoles utilisent le procédé suivant : l'algorithme (asymétrique) de Diffie-Hellman (dont nous verrons par quel procédé (génial) il permet d'échanger de façon sûre des clés de chiffrement symétrique classique), n'est utilisé qu'une fois, pour échanger des clés de chiffrement symétrique, qui peuvent alors être utilisées en confiance. Mais cela n'élimine pas le risque lié à l'attaque par interposition.

En fait nous sommes ramenés au sempiternel problème dont nous nous croyions débarrassés : comment établir une relation de confiance entre Anne et Franz, comment échanger des clés dignes de foi. Mais nous avons quand même accompli un progrès : cet échange de clés doit être certifié, mais il peut se faire au grand jour puisque les clés sont désormais publiques. Afin d'interdire les identités d'emprunt, donc l'interposition (*Man in the middle*), les clés publiques doivent être signées par une autorité supérieure, ce qui donne naissance à la notion d'infrastructure de gestion de clés, ou IGC (PKI en anglais), voir plus loin page 176.

Vulnérabilité des cryptosystèmes

L'importance économique et sociale des systèmes de chiffrement incite à se poser la question de leur vulnérabilité : méritent-ils la confiance que nous plaçons en eux ?

Récemment des articles ont exhibé des collisions avec l'algorithme SHA-1, ce qui prend en défaut les propriétés énoncées à la page 47 [117] [118] [107]. Avant les travaux de Mesdames Xiaoyun Wang et Yiqun Lisa Yin, une attaque par collision

contre SHA-1 demandait 2^{80} opérations ; elles ont fait descendre ce nombre à 2^{63}, ce qui reste considérable, et demanderait sans doute, pour être réalisé dans un délai compatible avec un objectif opérationnel, un calcul distribué à l'échelle planétaire. L'algorithme sera sans doute néanmoins renforcé.

Cet épisode illustre l'éternelle course aux armements entre les attaquants et les défenseurs, qui avait déjà lieu entre les fabricants d'épées et les fabricants de boucliers. Mais il suggère aussi que les vraies attaques criminelles n'empruntent pas la voie difficile du cassage de protocole, qui intéresse surtout (de façon bien légitime) les chercheurs en cryptographie. Les pirates recherchent les failles de réalisation, donc souvent de programmation, qui sont hélas assez courantes et d'une exploitation bien plus facile.

Le mirage de la biométrie

Il est des sujets que la presse grand public fait ressurgir périodiquement lorsqu'elle est à court de copie : cela s'appelle des marronniers. Parmi les marronniers informatiques figurent en tête les prétentions de l'« intelligence artificielle », avec au premier rang la traduction automatique et la reconnaissance vocale. Ces valeurs sûres ont été rejointes depuis quelques années par la *biométrie*, qui serait la panacée destinée à résoudre tous les problèmes de contrôle d'accès au SI. Et depuis que les passeports des États-Unis comportent des données biométriques, cette idée semble aller de soi.

Or cette idée est très discutable, sinon fausse, parce qu'elle repose sur une confusion entre *identification* et *authentification*. Pour reprendre les définitions de l'article de Philippe Wolf [124], « s'identifier, c'est communiquer son identité, s'authentifier, c'est apporter la preuve de son identité. » Les procédés biométriques sont mieux adaptés à l'identification qu'à l'authentification, pour les raisons exposées dans l'article cité ci-dessus, qui retrace ainsi le processus que pourrait suivre une authentification par de tels procédés :

Phase 1 présentation de la donnée biométrique par la personne à authentifier ;

Phase 2 acquisition de cette donnée par un lecteur biométrique ;

Phase 3 traitement de cette donnée par un dispositif électronique qui la transforme en une information numérique, sous forme d'un fichier ; ce codage peut faire appel à des techniques cryptographiques ;

Phase 4 comparaison de ce fichier caractérisant la personne à authentifier avec une donnée de référence (quand la personne s'est identifiée au préalable) ou avec des données préstockées de références (représentant l'ensemble des personnes que l'on souhaite authentifier) ;

Phase 5 décision, à partir de la comparaison effectuée en phase 4, d'authentifier ou non la personne grâce à une fonction mathématique ou statistique (on retrouve la définition initiale de la biométrie). Ici, la décision binaire (réponse par oui ou par non) est propagée (de manière sûre de préférence) au dispositif informatique demandant l'authentification.

Dans la réponse à une réponse à cet article (oui, le sujet est controversé), Robert Longeon [85] résume avec concision les inconvénients qui résultent d'un tel procédé : « Les réserves exprimées par l'article sur l'authentification biométrique proviennent du fait que l'authentifiant biométrique est une donnée publique (ce que n'est pas, par exemple, un bon mot de passe) et non révocable en cas de compromission (un mot de passe ou une clé se changent régulièrement). »

En effet, il est dans la nature d'un procédé d'authentification d'être exposé à la compromission, et lorsqu'il est compromis il faut pouvoir le *révoquer* : ainsi un mot de passe dont on soupçonne la divulgation doit être changé. Les dispositifs biométriques ne sont pas à l'abri de la compromission : si l'on considère les cinq phases énumérées ci-dessus, il sera peut-être difficile à un attaquant d'usurper l'empreinte digitale ou l'iris de l'œil de la personne dont il veut usurper l'identité, ce qui correspondrait à une attaque sur la phase 1, et certes de telles possibilités d'attaque existent bel et bien (l'attaquant offre à la victime une coupe de champagne convenablement instrumentée pour obtenir son empreinte digitale, ou même lui coupe le doigt), mais une attaque sur les phases 3 et 4 se résume à un simple vol de fichier, ce qui sera d'autant plus gênant que la victime pourra difficilement révoquer son iris ou son doigt.

Pour donner un exemple réel des difficultés qui peuvent résulter de l'utilisation à des fins d'authentification de données irrévocables, on peut rappeler qu'il y a quelques années certains établissements bancaires américains avaient l'habitude de demander au téléphone à leurs clients d'authentifier leur identité en donnant leur numéro de sécurité sociale et le nom de jeune fille de leur mère. Des escrocs se sont procuré ces données et les ont utilisées pour donner aux banques des ordres tout à leur avantage. La situation des clients était très embarrassante, parce qu'ils ne pouvaient changer ni de numéro de sécurité sociale, ni de mère : on avait utilisé un identifiant comme procédé d'authentification, erreur fatale mais fréquente.

3

Malveillance informatique

Parmi les multiples procédés d'attaque contre le système d'information, il convient de réserver une place spéciale (et un chapitre ici) à une famille de logiciels malveillants (les anglophones ont créé à leur intention le néologisme *malware*) qui se répandent en général par le réseau, soit par accès direct à l'ordinateur attaqué, soit cachés dans un courriel ou sur un site Web attrayant, mais aussi éventuellement par l'intermédiaire d'une disquette, d'une clé USB ou d'un CD-Rom. La destination de ces logiciels est de s'installer sur l'ordinateur dont ils auront réussi à violer les protections pour y commettre des méfaits, et aussi pour se propager vers d'autres victimes. Ce chapitre leur sera consacré ; essayons pour commencer d'en dresser une nomenclature.

Types de logiciels malveillants

Aujourd'hui, c'est un truisme, quiconque navigue sur l'Internet ou reçoit du courrier électronique s'expose aux logiciels malveillants que sont les virus, les vers et quelques autres que nous allons décrire. Comme tout le monde navigue sur l'Internet ou reçoit du courrier électronique, il importe que chacun acquière un mi-

nimum d'information sur ces logiciels nuisibles, ne serait-ce que pour pouvoir les nommer aux experts auxquels on demandera de l'aide pour s'en débarrasser. C'est l'objet du petit catalogue que voici.

Virus

Un virus est un logiciel capable de s'installer sur un ordinateur à l'insu de son utilisateur légitime. Le terme virus est réservé aux logiciels qui se comportent ainsi avec un but malveillant, parce qu'il existe des usages légitimes de cette technique dite de *code mobile* : les appliquettes Java et les procédures JavaScript sont des programmes qui viennent s'exécuter sur votre ordinateur en se chargeant à distance depuis un serveur Web que vous visitez, sans que toujours vous en ayez conscience, et en principe avec un motif légitime. Les concepteurs de Java et de JavaScript[1] nous assurent qu'ils ont pris toutes les précautions nécessaires pour que ces programmes ne puissent pas avoir d'effet indésirable sur votre ordinateur, bien que ces précautions, comme toutes précautions, soient faillibles. Les appliquettes Java s'exécutent dans un bac à sable *(sandbox)* qui en principe les isole totalement du système de fichiers qui contient vos documents ainsi que du reste de la mémoire de l'ordinateur.

En général, pour infecter un système, un virus agit de la façon suivante : il se présente sous la forme de quelques lignes de code en langage machine binaire qui se greffent sur un programme utilisé sur le système cible, afin d'en modifier le comportement. Le virus peut être tout entier contenu dans ce greffon, ou il peut s'agir d'une simple amorce, dont le rôle va être de télécharger un programme plus important qui sera le vrai virus.

Une fois implanté sur son programme-hôte, le greffon possède aussi en général la capacité de se recopier sur d'autres programmes, ce qui accroît la virulence de l'infection et peut contaminer tout le système ; la désinfection n'en sera que plus laborieuse.

On remarque que la métaphore par laquelle ce type de programme est nommé virus n'est pas trop fallacieuse, car les vrais virus, ceux de la biologie, procèdent de façon assez analogue : sans trop schématiser, on peut dire qu'un virus est un fragment d'acide désoxy-ribo-nucléique (ADN) ou d'acide ribo-nucléique (ARN)

[1]Incidemment, contrairement à ce que donnent à croire leurs noms, Java et JavaScript sont des langages qui n'ont rien à voir l'un avec l'autre.

dans le cas d'un rétrovirus, enveloppé dans une sorte de sachet qui lui permet de résister à l'environnement extérieur tant qu'il ne s'est pas introduit dans un organisme-hôte. Une fois qu'il a pénétré dans une cellule de l'hôte, ce fragment d'ADN ou d'ARN, dont on sait qu'il représente une sorte de « programme génétique », utilise la machinerie cellulaire de l'hôte pour se reproduire et envahir d'autres cellules, ce qui peut provoquer une maladie.

POUR EN SAVOIR PLUS

Pour ce parallèle entre informatique et biologie on pourra se reporter à un article de David Evans. « What Biology Can (and Can't) Teach Us About Security ». *USENIX Security Symposium*, 12 août 2004. `http://www.cs.virginia.edu/~evans/usenix04/usenix.pdf`.

Le problème que doit surmonter le virus informatique, comme son collègue biologique, c'est d'échapper au système immunitaire de l'hôte, qui cherche à le détruire, et comme en biologie les méthodes les plus efficaces pour atteindre ce but reposent sur les mutations et le polymorphisme, c'est-à-dire que le virus modifie sa forme, son aspect ou son comportement afin de ne pas être reconnu par son prédateur. Pour les virus informatiques comme pour ceux de la biologie, la stratégie de survie peut aussi comporter une *période d'incubation*, au cours de laquelle le malade ignore son état et peut contaminer son entourage, ainsi que des *porteurs sains,* propices également à la contagion. En effet, un virus qui tue trop rapidement sa victime, que ce soit au sens propre ou au sens figuré, limite par là-même ses capacités de propagation.

Virus réticulaire (botnet)

La cible d'un virus informatique peut être indirecte : il y a des exemples de virus qui se propagent silencieusement sur des millions d'ordinateurs connectés à l'Internet, sans y commettre le moindre dégât. Puis, à un signal donné, ou à une heure fixée, ces millions de programmes vont se connecter à un même serveur Web, ce qui provoquera son effondrement. C'est ce qu'on appelle un déni de service distribué *(Distributed Denial of Service, DDoS).*

Un tel virus s'appelle en argot SSI un *bot*, et l'ensemble de ces virus déployés un *botnet*. Les ordinateurs infectés par des bots sont nommés zombis.

De l'historique du terme *zombi*

Le terme *zombi* utilisé pour qualifier les ordinateurs qui ont été infectés par un virus (réticulaire, cheval de Troie, ...) ne doit bien sûr rien au hasard. Ces ordinateurs ressemblent effectivement à des morts-vivants : ces machines donnent l'impression de ne rien faire (il serait plus judicieux d'écrire que leur utilisateur n'a pas conscience de l'infection par un programme malveillant) et pourtant elles participent à de mauvaises actions, comme par exemple l'envoi de courrier électronique non sollicité ou bien des attaques par déni de service distribué.

De l'avis même des fournisseurs d'accès à l'Internet, ces postes de travail infectés représentent aujourd'hui la principale menace visible : une contribution même modeste (quelques dizaines de messages par heure) de chaque ordinateur, vu le nombre de machines infectées (des centaines de milliers rien qu'en France), suffit pour être à l'origine de dizaines de millions de messages non sollicités chaque jour.

Au grand dam des techniciens avertis qui sont lésés dans l'affaire, l'une des mesures phare adoptées (ou en cours d'adoption) par de nombreux fournisseurs d'accès à l'Internet grand public est de restreindre l'accès à l'Internet. Ainsi face aux *zombis* qui envoient des courriers non sollicités, le filtrage du port 25 devient une norme de fait, obligeant les utilisateurs avertis à prendre le risque de confier leur courrier électronique à leur fournisseur (l'utilisateur moins averti le fait probablement déjà et de façon naturelle).

Nous donnerons une définition plus complète du port à l'encadré page 121, mais pour l'instant il nous suffit de savoir que chaque extrémité d'un flux de données en circulation sur le réseau est identifiée par un numéro arbitraire mais unique, son numéro de port. Le port 25 est dévolu au protocole SMTP utilisé par le courrier électronique, et ainsi chaque émission de message électronique a pour destination le port 25 d'un serveur de messagerie.

Existe-t-il encore des virus qui ne soient pas des chevaux de Troie ?

La réponse à cette question est bien sûr oui et il faut continuer à lutter. Les chiffres montrent cependant qu'en termes de volume les virus circulant du fait de *zombis* ou de « chevaux de Troie » représentent bien plus de 95% du volume des messages avec virus circulant sur le réseau (chiffres obtenus d'un grand prestataire français d'accès à l'Internet, sur ses passerelles destinées au marché entreprise).

Ver

Un ver *(worm)* est une variété de virus qui se propage par le réseau. Il peut s'agir d'un *bot* (cf. ci-dessus). En fait, alors qu'il y a cinq ou six ans les virus n'étaient pas des vers (ils ne se propageaient pas par le réseau) et les vers n'étaient pas des virus

(ils ne se reproduisaient pas), aujourd'hui la confusion entre les deux catégories est presque totale.

Cheval de Troie

Un cheval de Troie *(Trojan horse)* est un logiciel qui se présente sous un jour honnête, utile ou agréable, et qui une fois installé sur un ordinateur y effectue des actions cachées et pernicieuses.

Porte dérobée

Une porte dérobée *(backdoor)* est un logiciel de communication caché, installé par exemple par un virus ou par un cheval de Troie, qui donne à un agresseur extérieur accès à l'ordinateur victime, par le réseau.

Bombe logique

Une bombe logique est une fonction, cachée dans un programme en apparence honnête, utile ou agréable, qui se déclenchera à retardement, lorsque sera atteinte une certaine date, ou lorsque surviendra un certain événement. Cette fonction produira alors des actions indésirées, voire nuisibles.

Logiciel espion

Un logiciel espion, comme son nom l'indique, collecte à l'insu de l'utilisateur légitime des informations au sein du système où il est installé, et les communique à un agent extérieur, par exemple au moyen d'une porte dérobée.

Une variété particulièrement toxique de logiciel espion est le *keylogger* (espion dactylographique ?), qui enregistre fidèlement tout ce que l'utilisateur tape sur son clavier et le transmet à son honorable correspondant ; il capte ainsi notamment identifiants, mots de passe et codes secrets.

Le *rootkit* de Sony

Nous ne saurions entreprendre ce tour d'horizon de la malfaisance sans évoquer une affaire où le scandale le dispute au ridicule.

Un *rootkit* est un programme ou un ensemble de programmes qui permettent à un pirate de maintenir durablement un accès frauduleux à un système informatique, généralement par l'ouverture de portes dérobées (cf. section 3 page précédente) et par des modifications vicieuses du système.

Le 31 octobre 2005, le spécialiste reconnu des systèmes Windows Mark E. Russinovich publiait sous le titre *Sony, Rootkits and Digital Rights Management Gone Too Far* un article[2] dans la revue en ligne *Sysinternals* où il racontait la mésaventure suivante.

Il testait sur son ordinateur le logiciel de détection d'intrusion *RootkitRevealer* (RKR), destiné comme son nom l'indique à détecter la présence de *rootkits*.

Que révéla RKR à Russinovich ? Un répertoire caché, plusieurs pilotes de périphériques cachés, et un programme caché. Russinovich, auteur notamment de l'ouvrage de référence *Windows Internals : Windows 2000, Windows XP & Windows Server 2003* [92], n'est pas précisément un utilisateur naïf et il applique des règles de sécurité scrupuleuses. Étonné de se voir ainsi piraté, il mobilisa toute sa science des structures internes de Windows et des outils d'analyse pour percer ce mystère (les détails sont exposés dans l'article cité en référence) : quelle ne fut pas sa surprise en découvrant que le *rootkit* incriminé était un logiciel commercial arborant fièrement la marque de la société qui l'avait développé, *First 4 Internet*. Cette société avait créé un ensemble de logiciels destinés à implémenter une technologie nommée XCP, dont la fonction est d'exercer des contrôles d'accès sur les CD musicaux enregistrés commercialisés selon les spécifications du protocole *Digital Rights Management* (DRM). *First 4 Internet* avait vendu sa technologie à plusieurs sociétés, dont Sony, et en constatant cela Russinovich se rappela avoir acheté peu de temps auparavant un CD Sony qui ne pouvait être joué qu'au moyen du logiciel inscrit sur le CD lui-même, et qui ne pouvait être recopié que trois fois. C'est ce que l'on appelle un CD au contenu protégé contre les copies.

En fait, lorsque le CD était joué sur un ordinateur, le logiciel inscrit sur le CD se recopiait dans le système, à l'insu de l'utilisateur. Une fois installé, il se comportait comme un logiciel espion, et envoyait à Sony l'identification du CD introduit dans le lecteur de l'ordinateur ; avec cet envoi , Sony était informé chaque fois qu'un CD donné était joué sur tel ou tel ordinateur, et recevait également l'adresse IP de cet ordinateur. De surcroît, ce logiciel assez mal conçu et réalisé créait dans le système des vulnérabilités supplémentaires qui facilitaient des attaques ultérieures par d'autres logiciels malfaisants. Clairement, le *Big Brother* du roman de George Orwell commençait à prendre réalité.

Mais le plus piquant (ou le plus scandaleux) de cette histoire, c'est que pour réaliser leur logiciel secret et malfaisant destiné à espionner leurs clients et à protéger de façon abusive leurs droits, *First 4 Internet* et son mandant Sony avaient purement et simplement piraté des parties de certains logiciels libres sous licence GPL dans des conditions contraires aux termes de cette licence, c'est-à-dire qu'ils n'avaient pas hésité à enfreindre les droits d'autrui.

Les formes de malveillance

Longtemps les actes de malveillance informatique tels que ceux que nous venons de décrire furent le plus souvent le fait de jeunes gens motivés par la recherche de la renommée parmi leurs collègues pirates (les *script kiddies*). Les auteurs de logiciels malveillants sont souvent dotés de compétences techniques élevées, nécessaires pour détecter et exploiter des vulnérabilités souvent subtiles ; ils mettent ensuite leurs logiciels à la disposition de la communauté, et des pirates peu qualifiés peuvent facilement les utiliser. Il est donc faux que tous les pirates soient des experts de haut niveau, la plupart sont des ignorants qui se contentent de lancer sur le réseau des logiciels nuisibles écrits par d'autres.

Cette malveillance « sportive » (et néanmoins criminelle) cède de plus en plus de terrain à une malveillance à but lucratif. Ainsi, les dénis de service distribués tels que celui que nous avons décrit au début de ce chapitre sont couramment utilisés contre des sites marchands pour exercer contre eux un chantage et en obtenir une rançon.

Michel Volle souligne dans son ouvrage *De l'Informatique* [115] que les pirates de l'informatique progressent plus vite que la recherche en sécurité, parce qu'ils utilisent les méthodes du logiciel libre, alors que la recherche publique a du mal à recruter et que la recherche des entreprises s'enferme dans un secret qui souvent ne sert qu'à dissimuler une certaine indigence. La compétence des pirates augmente, ainsi que le nombre et l'ingéniosité de leurs attaques, cependant que l'on ne compte aux États-Unis que 200 chercheurs en sécurité dans les universités et les entreprises, nous dit Michel Volle.

Si vous voulez aujourd'hui créer votre propre virus ou votre cheval de Troie, nul besoin d'être un virtuose du débordement de tampon, ni même de savoir programmer : vous trouverez sur le Web de magnifiques kits de développement avec des environnements graphiques à la dernière mode qui vous faciliteront le travail, il vous suffira de cliquer sur les boutons de votre choix, et le logiciel malfaisant sur mesure sera généré automatiquement par l'outil. Vous disposerez également du système de lancement sur l'Internet, et tout cela gratuitement. Dans ces conditions, ce qui est étonnant, c'est qu'il n'y ait pas plus de dégâts, ou du moins de dégâts patents.

Courrier électronique non sollicité (spam)

Le courrier électronique non sollicité *(spam)* consiste en « communications électroniques massives, notamment de courrier électronique, sans sollicitation des destinataires, à des fins publicitaires ou malhonnêtes », selon Wikipédia. Ce n'est pas à proprement parler du logiciel, mais les moyens de le combattre sont voisins de ceux qui permettent de lutter contre les virus et autres malfaisances, parce que dans tous les cas il s'agit finalement d'analyser un flux de données en provenance du réseau pour rejeter des éléments indésirables.

Les messages électroniques non sollicités contiennent généralement de la publicité, le plus souvent pour de la pornographie, des produits pharmaceutiques destinés à améliorer les dimensions et les performances de certaines parties du corps humain, des produits financiers ou des procédés d'enrichissement rapide. Parfois il s'agit d'escroqueries pures et simples, qui invitent le lecteur à accéder à un site qui va lui extorquer son numéro de carte bancaire sous un prétexte plus ou moins vraisemblable, cela s'appelle le *phishing*. Rappelons la définition de l'escroquerie par les articles L 313-1 à 313-3 du Code Pénal : « Le fait, soit par l'usage d'un faux nom, soit par l'abus d'une qualité vraie, soit par l'emploi de manœuvres frauduleuses, de tromper une personne physique et de la convaincre à remettre des fonds, des valeurs ou un bien quelconque ou à fournir un service ou à consentir un acte opérant obligation ou décharge ».

Attaques sur le Web et sur les données

Avec la multiplication et la diversification des usages du Web, notamment pour des sites marchands ou de façon plus générale pour des transactions financières, sont apparues de nouveaux types d'attaques qui en exploitent les faiblesses de conception. D'autre part, des attaquants que l'on pourrait nommer les charognards informatiques exploitent dans les documents des zones que leurs auteurs croient avoir effacées, cependant que d'autres pillent les disques durs des matériels de rebut.

Injection SQL

L'attaque par *injection SQL* vise les sites Web qui proposent des transactions mal construites dont les résultats sont emmagasinés dans une base de données relationnelle. Elle consiste en ceci : SQL est un langage qui permet d'interroger et de mettre à jour une base de données relationnelle ; les requêtes sont soumises au moteur de la base en format texte, sans être compilées. Une requête typique est construite à partir de champs de formulaire remplis par l'internaute. Si l'auteur du site a été paresseux, il aura construit ses requêtes en insérant directement les textes rédigés par l'internaute, sans en contrôler la longueur ni le format ni le contenu. Ainsi, un utilisateur malveillant informé de cette faille (ou qui la soupçonnerait) peut confectionner un texte tel qu'une fois incorporé à une requête SQL il ait des effets indésirables sur la base de données, par exemple en y insérant directement des ordres du langage, de telle sorte qu'ils soient interprétés. En voici un exemple ; l'instruction suivante construit directement à partir du nom introduit par l'utilisateur une requête SQL innocente, qui extrait de la base des utilisateurs tous les enregistrements qui concernent celui-là en particulier :

```
requete := "SELECT * FROM clients
                WHERE nom = '" + nom_client + "';"
```

Soit un attaquant informé de cette faille, qui au lieu d'entrer dans le formulaire un nom valide, introduit la chaîne de caractères suivante :

*x'; DROP TABLE clients ; SELECT * FROM data WHERE nom LIKE '%" as "nom_client*

La requête sera :

```
SELECT * FROM clients WHERE nom = 'x'; DROP TABLE clients;
       SELECT * FROM secrets WHERE nom LIKE '%';
```

avec, comme résultat, la destruction pure et simple de la table `clients` et un accès imprévu à la table `secrets`, dont le nom suggère qu'elle n'est pas destinée à être lue par les internautes.

La parade à ce type d'attaque consiste essentiellement à écrire des programmes moins naïfs, qui vérifient les données introduites par les utilisateurs avant de les utiliser, et en particulier qui éliminent les caractères qui ont une valeur sémantique spéciale pour SQL. Cette recommandation vaut d'ailleurs pour *tous* les programmes.

Cross-site scripting

Cette famille d'attaques, pour laquelle je n'ai pas trouvé de traduction française généralement admise, est apparue avec le langage *JavaScript*, dont l'usage principal est d'insérer dans une page en HTML sur le Web un programme qui viendra s'exécuter dans le navigateur de l'internaute. Sur une page vulnérable (elles sont légion), un pirate pourra installer un programme *JavaScript* furtif, qui pourra accomplir des actions sur l'ordinateur de l'internaute à son insu. Ces actions risquent d'être peu désirables, mais surtout elles peuvent rediriger discrètement la navigation vers un site malveillant qui pourra injecter du code dans la page visitée. Il convient de ne pas sous-estimer les risques induits par ce genre de faille, par exemple si la page détournée comporte des demandes d'authentification avec mot de passe ou des transactions financières.

Palimpsestes électroniques

Au moyen-âge le parchemin sur lequel étaient copiés les manuscrits était rare et cher : aussi on grattait les livres passés de mode pour en réutiliser le parchemin et y recopier les derniers succès de librairie. Les chercheurs modernes ont réussi à lire le texte gratté sur de tels manuscrits, appelés *palimpsestes*.

Le XXe siècle a aussi ses palimpsestes : si l'on n'y prend garde, les fichiers produits par *Microsoft Word* conservent fidèlement dans un coin la trace des modifications passées, et un lecteur habile peut les retrouver. Le Premier ministre britannique Tony Blair a été victime de cet artifice : son cabinet a publié sur le site Web gouvernemental un document *Word* d'explications relatif à l'engagement britannique dans la guerre en Irak, et des journalistes malicieux ont reproduit l'historique de l'argumentation, qui a fait scandale.

Matériels de rebut

Au début de l'année 2003 un petit article [1] a suscité un certain étonnement : un chercheur et un étudiant du MIT, à Cambridge dans le Massachusetts, Simson Garfinkel et Abhi Shelat, ont acheté 158 disques durs d'occasion, souvent considérés comme des épaves, sur des sites d'enchères en ligne tels que *eBay*. Ils ont entrepris de les lire et d'en analyser le contenu ; 129 disques étaient en état de marche et lisibles, sur seulement 12 les données avaient été convenablement effacées ; sur 28 disques aucune manœuvre d'effacement n'avait été entreprise. Lorsque des

opérations d'effacement avaient été effectuées, elles avaient souvent été inefficaces : en effet la destruction d'un fichier, ou même le formatage du disque, ne remet pas effectivement chaque bloc du disque à zéro. Sur un des disques soi-disant formatés, Garfinkel et Shelat ont trouvé 5 000 numéros de cartes de crédit. De grandes quantités de données personnelles financières ou médicales, ainsi que de courrier privé et de pornographie, ont été découvertes. Que cela serve de leçon à quiconque met un ordinateur au rebut ! Il est vivement conseillé de détruire soigneusement tous les supports de données qui ont été utilisés pour des usages sensibles.

Lutte contre les malveillances informatiques

La prolifération des formes de malveillance informatique s'accomplit parallèlement à la convergence de leurs méthodes : néanmoins l'utilisateur n'est pas sans défense contre les attaques de plus en plus nombreuses et de plus en plus puissantes, il existe des armes défensives. Nous examinerons ici la plus nécessaire : le logiciel antivirus. Plus loin dans cet ouvrage nous étudierons le pare-feu et les systèmes de détection et de prévention des intrusions.

Ce que l'on peut dire de la lutte contre les virus s'applique aussi dans une large mesure à la lutte contre les autres malfaisances informatiques, car aujourd'hui les logiciels malfaisants sont très polyvalents : la plupart des virus sont aussi des vers, qui ouvrent des portes dérobées et pratiquent l'espionnage pour « améliorer » leurs performances. Ils peuvent aussi à l'occasion émettre du courrier non sollicité et se procurer des numéros de cartes bancaires.

Antivirus

Ces sections sont notamment inspirées par certaines informations et analyses contenues dans un article d'Éric Filiol [53]. Pour une étude approfondie il faudra aussi se reporter à son livre consacré aux virus [51].

Il existe des logiciels dits *antivirus*, qui peuvent s'installer principalement en deux sortes d'endroits :

- soit à l'entrée d'un réseau local, là où arrivent les flux en provenance de l'Internet ; certains de ces flux seront filtrés pour y détecter des virus, essentiellement les flux relatifs aux protocoles SMTP (courrier électronique) et HTTP (Web) ;

- soit sur le poste de travail de l'utilisateur, et là l'antivirus servira généralement à inspecter et désinfecter le disque dur (il convient de garder à l'esprit que certains virus s'exécutent en mémoire vive, sans s'enregistrer sur le disque).

Il y a essentiellement deux modes de fonctionnement des logiciels antivirus :

- mode statique : le logiciel est activé uniquement sur ordre de l'utilisateur, par exemple pour déclencher une inspection du disque dur ;
- mode dynamique : le logiciel est actif en permanence, et il scrute certains événements qui surviennent dans le système, ce qui induit une consommation non négligeable de ressources telles que temps de processeur et mémoire, mais permet une meilleure détection des attaques, notamment par analyse comportementale des logiciels suspects d'être contaminés.

Où mettre des antivirus ?

Le lecteur l'aura remarqué dans l'exposé : certains systèmes sont plus menacés par le risque viral que d'autres. Est-ce parce que ces systèmes d'exploitation sont moins bien sécurisés que les autres ? L'auteur de ces lignes n'a pas la prétention de répondre à cette question. Ce qui est en revanche certain, c'est que la popularité d'un système d'exploitation ou d'un logiciel attire naturellement à lui les auteurs de codes malveillants : quel serait l'intérêt, pour la gloire de l'auteur anonyme, d'écrire un virus qui exploiterait un défaut dans un logiciel qui n'est utilisé qu'à dix exemplaires dans le monde ?

Tout utilisateur se doit donc de mettre en œuvre une protection contre les codes malveillants si son ordinateur est concerné par le risque... et il faudra en inclure le coût dans le budget global de fonctionnement de l'ordinateur. Les entreprises d'une certaine taille l'ont compris et intègrent ce risque dans la gestion de leur parc micro-informatique.

Ces dispositifs sont souvent complétés par des solutions centralisées et indépendantes du poste de travail : les serveurs de messagerie, les mandataires permettant d'accéder au Web, ... disposent parfois eux aussi de solutions de recherche et d'élimination de virus, voire plus largement de codes malicieux ou supposés tels.

Installer simultanément un logiciel antivirus sur le poste de travail et un contrôle central sur les passerelles, ce sont des précautions complémentaires : chacune apporte sa brique pour réduire le risque (on ne parlera pas de l'éliminer, le risque zéro n'existe pas).

Les passerelles centralisées présentent l'avantage d'avoir un point central de contrôle des flux en provenance de l'extérieur, point central auquel l'entreprise peut mettre en œuvre les moyens nécessaires pour être à jour et au fait des menaces les plus récentes et ainsi protéger son parc informatique. Ces passerelles n'apportent cependant pas de solution à certaines situations, par exemple :

- l'utilisateur nomade *(road warrior)* qui pour les besoins de son travail doit accéder à l'Internet sans pour autant être en mesure d'ouvrir un accès distant de type

VPN avec son réseau d'entreprise (où il bénéficierait alors des solutions de sécurité centrales) ;

- les contenus chiffrés ne sont par définition pas analysables par les passerelles centrales (celles-ci pourraient être configurées pour rejeter ce type de contenu, mais dans ce cas il faut bien avoir conscience des effets induits indésirables) ;
- l'analyse de formats inconnus — un antivirus en position centrale saura-t-il reconnaître tous les formats de fichiers de la planète ? Certainement pas !.

C'est à ce stade que les solutions de poste de travail apportent un avantage : les solutions modernes vont bien au-delà de l'analyse des fichiers présents sur le disque dur de l'ordinateur. Ces solutions de sécurité vont intercepter, en temps réel, certains accès à des fichiers, des opérations spécifiques et faire leur travail de recherche d'un code malicieux : la passerelle centrale laissera ainsi peut-être passer un virus dans un message chiffré S/MIME, l'antivirus de poste de travail le détectera probablement après déchiffrement, et surtout lorsque l'utilisateur tentera d'enregistrer le message déchiffré sur son disque dur.

La principale difficulté de l'antivirus de poste de travail réside en sa gestion, et notamment la mise à jour des moteurs de recherche de code malicieux et des bases de signature. Des systèmes existent bien, mais en général les mises à jour sont moins fréquentes qu'en central (imaginez l'impact de 100 000 ordinateurs d'une grande entreprise qui vont vérifier toutes les cinq minutes s'il y a des mises à jour) et la période de risque pour la prise en compte d'une menace récente est légèrement plus longue qu'avec une solution en central.

L'auteur de ces lignes tient à préciser qu'il n'a aucun lien avec l'industrie fort lucrative des éditeurs de solutions antivirales, et, comme tout utilisateur final, il est fort mécontent des tarifs pratiqués, mais il n'a vraiment pas le choix : le risque est aujourd'hui bien trop grand.

Les techniques de détection

Éric Filiol distingue trois familles de procédés de détection des virus : l'analyse de forme, le contrôle d'intégrité et l'analyse dynamique de comportement.

L'*analyse de forme* consiste à détecter la présence d'un virus dans un fichier par des caractères statiques qui permettent de le reconnaître. Éric Filiol distingue les éléments suivants :

- La recherche de signatures : on cherche un motif textuel, c'est-à-dire une suite de bits, caractéristique d'un virus connu. Cette méthode ne permet pas de détecter un nouveau virus, ni un virus déjà connu mais modifié. Elle impose l'installation et la mise à jour en permanence d'une base de données

des signatures. Quelques heures de retard dans la mise à jour peuvent suffire à mettre en échec la protection.

- L'analyse spectrale : certaines instructions sont rares dans les programmes ordinaires mais fréquentes dans les virus, ainsi une analyse statistique de la fréquence des instructions peut permettre la détection de virus, y compris parfois de virus inédits. Cette méthode est sujette aux faux positifs, c'est-à-dire à la détection, à tort, d'un virus dans un fichier exécutable légitime.
- L'analyse heuristique : il s'agit d'établir et de mettre à jour un corpus de règles qui permettent de caractériser les propriétés d'un fichier suspect. Cette méthode, comme la précédente, est sujette aux faux positifs.

Le ***contrôle d'intégrité*** consiste à détecter la modification anormale d'un fichier, qui peut signaler sa contamination par un virus. Pour mettre en œuvre cette méthode, il faut calculer pour chaque fichier sensible une empreinte numérique infalsifiable par une fonction de condensation (on dit aussi hachage, voir page 47). Constituer et mettre à jour une base de données de telles empreintes est difficile pratiquement, et cette méthode est de moins en moins utilisée.

L'***analyse dynamique de comportement*** consiste à scruter les actions d'un programme dès qu'il s'exécute et à détecter les activités suspectes : tentatives d'accès en écriture à des fichiers de programmes exécutables, ou à des bibliothèques, ou à des zones du disque réservées au système.

Aucune de ces méthodes n'est infaillible, aussi convient-il d'avoir recours à une combinaison de méthodes ; l'inconvénient est qu'un logiciel ainsi constitué devient encombrant, lent, et il ralentit le système, ce qui risque de dissuader l'utilisateur. Il est néanmoins nécessaire de déployer des antivirus sur le poste de travail *et* en entrée de réseau, ce qui permettra d'éviter la plupart des infections. Ensuite, comme on sait que certaines infections franchiront les barrières, il faut faire en sorte d'en limiter les conséquences, notamment en fractionnant son réseau en segments isolés les uns des autres pour réduire l'ampleur d'une éventuelle contamination.

Des virus blindés pour déjouer la détection

Le point commun entre les trois procédés de détection de virus dont nous avons emprunté ci-dessus la nomenclature à Éric Filiol, c'est que pour élaborer un antivirus le virologue a pu se procurer un exemplaire du virus et en analyser le texte.

Le même auteur signale dans un autre article [52] la possibilité, démontrée expérimentalement, de créer des virus furtifs ou, pour reprendre sa terminologie, blindés, qu'il est pratiquement impossible d'isoler sous leur forme virulente.

Le principe en est cryptologique : pendant le transport, le texte du virus est chiffré. Le succès de l'attaque repose sur l'impossibilité pour le défenseur de se procurer la clé de déchiffrement. Cela semble paradoxal, car pour attaquer il faudra bien que le virus soit déchiffré, et donc que la clé de déchiffrement soit d'une façon ou d'une autre accessible depuis le site de la cible.

De tels virus sont concevables pour des attaques ciblées : le code du virus comporte une procédure de déchiffrement qui réunit, avant de déchiffrer le texte du code virulent, des données d'activation, les unes présentes sur le site visé, les autres fournies à distance par l'attaquant ; la clé de déchiffrement est construite à partir de ces données d'activation, supposées impossibles à reproduire. Après avoir commis l'attaque, le virus s'auto-désinfecte, ainsi qu'en cas d'échec d'une des étapes de son activité, ce qui rend pratiquement impossible à un analyste de s'en procurer un exemplaire. Éric Filiol a démontré que, même en possession du texte du virus blindé expérimental construit par des virologues, l'analyse en demanderait 2^{512} opérations, ce qui revient à la déclarer impossible dans les conditions techniques de ce jour.

Si les virus construits selon de tels principes n'ont jusqu'à présent provoqué que des dégâts limités, c'est à cause de leur réalisation maladroite : algorithmes cryptographiques mal choisis et mal implémentés, mauvaise gestion des clés.

Quelques statistiques

Nous emprunterons ici quelques données quantitatives au *Rapport Sophos 2005 sur la gestion des menaces à la sécurité* [104], qui émane d'un grand éditeur d'antivirus, ainsi qu'aux études classiques du groupe IDC.

Sur le champ de bataille de la malfaisance informatique, le vandalisme ludique cède de plus en plus de terrain à la criminalité organisée. Les attaques virales massives, trop vite repérées, se voient remplacées par des infections de basse intensité, qui échappent à l'attention.

D'après le rapport cité en référence, 1 message électronique sur 44 comporte une charge virale ; à la fin novembre 2005, période marquée par l'épidémie due au ver Sober-Z, cette proportion a atteint 1 sur 12.

En décembre 2005, la base de données de virus de Sophos recense 114 000 virus, vers, chevaux de Troie et autres logiciels malfaisants, dont 15 907 apparus en 2005.

En ce qui concerne les méthodes de propagation des logiciels malfaisants, la connexion directe par le réseau représente 66,8%, les pièces jointes 15,1%, le *chat* (bavardage en direct par le réseau) 9,2%, le poste à poste (P2P) 4,5%, la navigation sur le Web 2,4%.

L'immense majorité des attaques visent des postes de travail équipés du système Windows de Microsoft. Les attaques contre les téléphones mobiles restent relativement peu nombreuses.

Selon le site de l'éditeur d'antivirus Sophos, un ordinateur équipé d'un système d'exploitation pour le grand public, connecté à l'Internet et dépourvu de protection (pare-feu, antivirus) est exposé à un risque de contamination qui atteint 40% au bout de dix minutes, 95% au bout d'une heure. Les observations personnelles de l'auteur ne démentent pas cette estimation.

Deuxième partie

Science de la sécurité du système d’information

4

La clé de voûte : le chiffrement

Ici nous entrons au cœur du sujet : les principes du chiffrement sur lesquels reposent toutes les techniques de sécurité informatique, et notamment les inventions révolutionnaires de l'échange public de clés et du chiffrement asymétrique. Il est surprenant que les bases mathématiques de ces inventions des années 1970 aient été connues des mathématiciens du XVIII[e] siècle comme Euler, et le lecteur découvrira qu'il peut les aborder avec la seule maîtrise de l'addition et de la multiplication.

Nous ne saurions tracer ici une histoire complète des codes secrets, pour laquelle le lecteur pourra se reporter au livre de Simon Singh [103] par exemple. Tout ce qui est antérieur à 1970 a un intérêt essentiellement historique, d'ailleurs passionnant et riche d'enseignements, ainsi le rôle récemment mis en lumière d'Alan Turing dans le déroulement de la Seconde Guerre mondiale, dont nous parlerons plus loin dans ce chapitre.

Avertissement

Ce chapitre reprend, en le développant, une part importante du contenu du chapitre 7 de mon livre *Les systèmes d'exploitation des ordinateurs – Histoire, fonctionnement, enjeux*, publié aux Éditions Vuibert, avec l'aimable autorisation de l'éditeur.

Le lecteur qui jugerait le contenu ce chapitre trop technique, notamment des pages 76 à 85, pourra s'en dispenser en première lecture, il pourra comprendre le reste de l'ouvrage, c'est promis ! Qu'il sache néanmoins que ces passages mathématiques constituent la garantie de la véracité de l'ensemble de l'ouvrage. *Nous encourageons le lecteur rigoureux à s'y plonger, tôt ou tard.*

Chiffrement symétrique à clé secrète

De l'époque de Jules César à la fin des années 1970, un grand nombre de systèmes de chiffrement ont été inventés, qui consistaient à faire subir à un texte clair une transformation plus ou moins complexe pour en déduire un texte inintelligible, dit *chiffré*. La transformation repose sur deux éléments : une fonction mathématique (au sens large) et une clé secrète. Seule une personne connaissant la fonction et possédant la clé peut effectuer la transformation inverse, qui transforme le texte chiffré en texte clair. C'est la même clé qui sert au chiffrement et au déchiffrement, et pour cette raison elle doit rester secrète : nous décrirons plus loin des systèmes de *chiffrement asymétrique*, qui utilisent des clés différentes pour le chiffrement et le déchiffrement, ce qui permet de rendre publique la clé de chiffrement, puisqu'elle ne permet pas le déchiffrement.

La science de l'invention des codes secrets s'appelle la cryptographie. La science, adverse, du déchiffrement de ces codes est la cryptanalyse. Si le cryptanalyste ignore tout de la fonction de chiffrement et de la clé, il aura le plus grand mal à déchiffrer, mais un bon code doit résister à la découverte de sa fonction de chiffrement tant que la clé reste secrète.

Une bonne fonction de chiffrement doit éviter de prêter le flanc à la cryptanalyse. Ainsi le code de César, qui reposait sur une simple transposition circulaire des lettres de l'alphabet, est très facile à décoder par l'analyse des fréquences des lettres dès lors que l'on sait dans quelle langue a été écrit le message. Un bon code doit aussi chiffrer de façons différentes deux occurrences successives d'un même texte

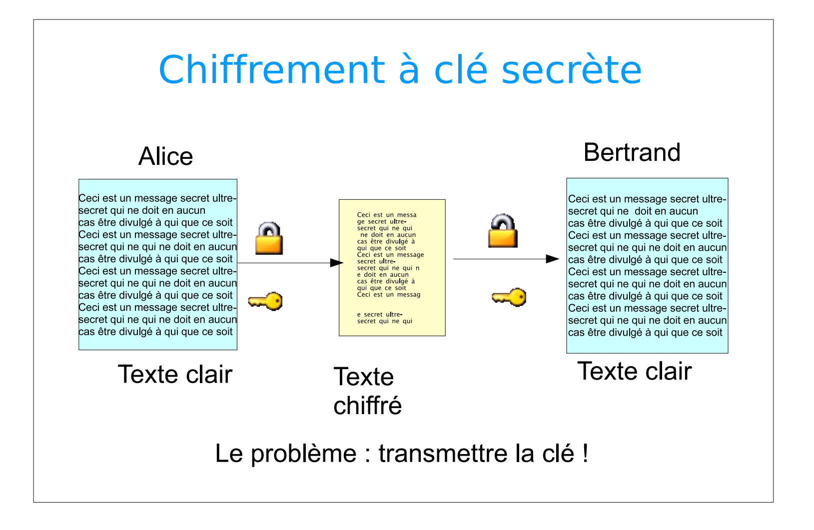


Figure 4.1
Principe du chiffrement symétrique

dans le corps du message pour éviter que la détection d'une répétition ne fournisse des indices au cryptanalyste. La connaissance simultanée d'un texte clair et de sa version chiffrée, comme dans le cas de Champollion et de la pierre de Rosette, est une aubaine pour le décodeur, comme l'occurrence de noms propres, etc.

Naissance de la cryptographie informatique : Alan Turing

L'invention de l'ordinateur a bien sûr donné un essor considérable à la cryptographie et à la cryptanalyse. Ce n'est d'ailleurs pas un hasard si le créateur du modèle théorique de l'ordinateur, Alan Turing, a été aussi pendant la guerre un formidable concepteur de machines à déchiffrer les codes allemands chiffrés par les automates *Enigma*. Les machines de Turing, appelées *Bombes*, étaient fondées sur une réalisation originale du logicien polonais Marian Rejewski. La courbe qui trace le succès des attaques de sous-marins allemands contre les convois transatlantiques qui acheminaient les fournitures américaines à la Grande-Bretagne subit des fluctuations importantes qui correspondent au délai à l'issue duquel l'équipe d'Alan Turing à Bletchley Park en Angleterre parvenait à déchiffrer plus ou moins parfaitement le code allemand après un changement de combinaison des *Enigma*. Lorsque l'on sait l'importance militaire qu'ont eue ces fournitures, on ne saurait sous-estimer la contribution de Turing à la victoire alliée.

Data Encryption Standard (DES)

Le premier système de chiffrement informatique normalisé fut créé par un Allemand émigré aux États-Unis en 1934, Horst Feistel. Sa nationalité et son métier de cryptographe lui valurent quelques difficultés avec la National Security Agency (NSA), désireuse avant tout de garder la maîtrise des moyens de chiffrement et de pouvoir percer les codes utilisés par des personnes privées. Finalement il mit ses compétences au service d'IBM, pour qui il développa au début des années 1970 le cryptosystème *Lucifer*, base du futur *Data Encryption Standard (DES)*.

Le DES repose sur les principes suivants : le texte clair est codé en numération binaire et découpé en blocs de 64 bits. Chaque bloc est découpé en demi-blocs dont les bits subissent des permutations complexes, puis les demi-blocs sont additionnés et soumis à d'autres transformations. L'opération est recommencée seize fois. La fonction de transformation comporte des variations en fonction de la clé, qui est un nombre arbitraire choisi par les utilisateurs du code. Le nombre de valeurs possibles pour la clé détermine le nombre de façons dont un message peut être chiffré. L'émetteur du message secret le chiffre selon l'algorithme DES au moyen de la clé, le destinataire applique la fonction inverse avec la même clé pour le déchiffrer.

La NSA a obtenu que la normalisation du DES en 1976 comporte une limitation de la taille de la clé à 56 bits, ce qui correspond à 10^{17} valeurs possibles. Aujourd'hui cette valeur est notoirement trop faible, et l'on utilise le triple DES, avec une longueur de clé de 112 bits.

La nouvelle norme AES *(Advanced Encryption Standard)* utilise des clés de 128, 192 et 256 bits. La mise en concurrence pour AES a été lancée le 2 janvier 1997 et le choix de la solution a eu lieu le 3 octobre 2000. C'est l'algorithme *Rijndael* développé par Joan Daemen et Vincent Rijmen de l'université catholique de Louvain qui a été retenu.

La postérité actuelle du DES procure un chiffrement qui peut être considéré comme robuste, à condition que soit résolu le problème crucial de tous les systèmes qui reposent sur une clé secrète utilisée aussi bien pour le chiffrement que pour le déchiffrement : les participants doivent s'échanger des clés de façon secrète, ce qui n'est pas simple.

Diffie et Hellman résolvent l'échange de clés

Si Alex veut entretenir une correspondance secrète avec Bérénice, ils peuvent convenir de chiffrer leurs messages avec un protocole tel que le triple DES, que nous venons de présenter. Ce protocole présente toutes les garanties de robustesse, mais il faudra que Bérénice et Alex conviennent d'une clé secrète : pour ce faire, ils devront se rencontrer, ce qui peut être impossible, ou se communiquer la clé par la poste : dans les deux cas, l'instant de l'échange est celui dont un espion peut profiter pour dérober leur secret et ainsi réduire à néant la sûreté de leurs communications. C'est le problème de l'échange de clés.

Le problème de l'échange de clés

Depuis des siècles le problème de l'échange des clés était considéré comme un inconvénient naturel du chiffrement. Les ambassades et les états-majors y consacraient des efforts importants, que les espions s'efforçaient de déjouer.

Avec l'utilisation de l'ordinateur et des télétransmissions, et la dématérialisation de l'information qu'ils autorisent, le problème se pose différemment. Dans les années 1970 un chercheur indépendant et excentrique, Whitfield Diffie, réfléchissait au moyen pour deux utilisateurs du réseau ARPANET d'échanger des courriers électroniques chiffrés sans se rencontrer physiquement au préalable pour convenir de la clé de chiffrement qu'ils utiliseraient. En 1974 il donna une conférence sur le sujet au centre de recherche Thomas J. Watson d'IBM à Yorktown Heights (déjà le lieu de travail de Horst Feistel), et là il apprit que Martin Hellman, professeur à l'université Stanford à Palo Alto, avait donné une conférence sur le même sujet. Aussitôt il prit sa voiture et traversa le continent pour rencontrer Hellman.

Un peu d'histoire

En 1969 l'ARPA *(Advanced Research Projects Agency)*, agence du ministère américain de la Défense pour la recherche, impulsa la création du réseau ARPANET pour faciliter les échanges entre les différents laboratoires de recherche avec lesquels elle avait des contrats. ARPANET fut l'ancêtre de l'Internet.

Diffie et Hellman cherchaient une méthode pour convenir d'un secret partagé sans le faire circuler entre les participants, en d'autres termes une fonction mathématique telle que les participants puissent échanger des informations dont eux

seuls pourraient déduire le secret. Les caractéristiques souhaitées d'une telle fonction sont la relative facilité de calcul dans le sens direct, et la quasi-impossibilité de calculer la fonction réciproque. Ainsi, si s est le secret en clair, F la fonction de chiffrement, c le secret chiffré, D la fonction de déchiffrement, il faut que $c = F(s)$ soit facile à calculer, mais $s = D(c)$ pratiquement impossible à calculer pour tout autre que les participants — au prix de quel stratagème, c'est ce que nous allons voir.

Fondements mathématiques de l'algorithme Diffie-Hellman

La solution au problème que se posaient Diffie et Hellman repose sur un chapitre de l'arithmétique très utilisé par les informaticiens, l'*arithmétique modulaire*, soit l'arithmétique fondée sur les classes d'équivalence *modulo* n.

Considérons l'ensemble des entiers relatifs $\mathbb{Z}$ muni de l'addition et de la multiplication. La division entière de a par b que nous avons apprise à l'école primaire y est définie ainsi :

$$a \div b \rightarrow a = b \times q + r$$

où q est le quotient et r le reste de la division. Ainsi :

$$13 \div 3 \rightarrow 13 = 3 \times 4 + 1$$

Intéressons-nous maintenant à tous les nombres qui, divisés par un nombre donné n, par exemple 3, donnent le même reste r. Nous avons déjà trouvé un nombre, 13, pour lequel $r = 1$; donnons-en quelques autres :

$$\begin{array}{rcl} 1 \div 3 & \rightarrow & 3 \times 0 + 1 \\ 4 \div 3 & \rightarrow & 3 \times 1 + 1 \\ 7 \div 3 & \rightarrow & 3 \times 2 + 1 \\ 10 \div 3 & \rightarrow & 3 \times 3 + 1 \\ 13 \div 3 & \rightarrow & 3 \times 4 + 1 \\ 16 \div 3 & \rightarrow & 3 \times 5 + 1 \end{array}$$

On dit que ces nombres constituent une classe d'équivalence, et qu'ils sont tous équivalents à $1 \bmod 3$ (prononcer « un modulo trois »), ce qui s'écrit :

$$4 \equiv 1 \bmod 3$$
$$7 \equiv 1 \bmod 3$$
$$\ldots$$

On construit de la même façon une classe des nombres équivalents à $0 \bmod 3$, qui contient $-6, -3, 0, 3, 6, 9, 12, \ldots$, et une classe des nombres équivalents à $2 \bmod 3$, avec $-7, -4, -1, 2, 5, 8, 11, \ldots$.

On peut définir une addition modulaire, par exemple ici l'addition mod 3 :

$$\begin{aligned} 4 + 7 \pmod 3 &= (4+7) \bmod 3 \\ &= 11 \bmod 3 \\ &= 2 \bmod 3 \end{aligned}$$

On démontre (exercice laissé au lecteur) que l'ensemble des classes d'équivalence *modulo* n muni de cette relation d'équivalence (réflexive, transitive) et de cette addition qui possède les bonnes propriétés (associative, commutative, existence d'un élément neutre $0 \bmod n$ et d'un symétrique pour chaque élément) possède une structure de groupe appelé groupe additif $\mathbb{Z}_n$ (prononcé « Z modulo n »).

On peut aussi faire des multiplications :

$$\begin{aligned} 4 \times 7 \pmod 3 &= (4 \times 7) \bmod 3 \\ &= 28 \bmod 3 \\ &= 1 \bmod 3 \end{aligned}$$

Nous pouvons montrer là aussi que la multiplication modulo 3 possède toutes les bonnes propriétés qui font de notre ensemble de classes d'équivalence un groupe pour la multiplication, mais cela n'est vrai que parce que 3 est premier. En effet si nous essayons avec les classes d'équivalence modulo 12, nous aurons des diviseurs de zéro, ce qui détruit la structure de groupe :

$$\begin{aligned} 4 \times 7 \pmod{12} &= (4 \times 7) \bmod 12 \\ &= 28 \bmod 12 \\ &= 4 \bmod 12 \end{aligned}$$

$$\begin{aligned} 4 \times 6 \pmod{12} &= (4 \times 6) \bmod 12 \\ &= 24 \bmod 12 \\ &= 0 \bmod 12 \end{aligned}$$

Dans la seconde expression, le produit de 4 et 6 est nul, ce qui est très regrettable. Aussi pourrons-nous bien définir un groupe multiplicatif $\mathbb{Z}_n^*$, qui si n est premier aura les mêmes éléments que le groupe additif $\mathbb{Z}_n$ à l'exclusion de 0, mais si n n'est pas premier il faudra en retrancher les classes correspondant aux diviseurs de n et à leurs multiples :

$$\begin{aligned} \mathbb{Z}_3^* &= \{1,2\} \\ \mathbb{Z}_{12}^* &= \{1,5,7,11\} \\ \mathbb{Z}_{15}^* &= \{1,2,4,7,8,11,13,14\} \end{aligned}$$

Dans ce groupe multiplicatif chaque élément a un inverse (sinon ce ne serait pas un groupe) :

$$\begin{aligned} 5 \times 5 \bmod 12 &= 25 \bmod 12 \\ &= 1 \bmod 12 \\ 7 \times 7 \bmod 12 &= 49 \bmod 12 \\ &= 1 \bmod 12 \\ 11 \times 11 \bmod 12 &= 121 \bmod 12 \\ &= 1 \bmod 12 \\ \\ 7 \times 13 \bmod 15 &= 91 \bmod 15 \\ &= 1 \bmod 15 \end{aligned}$$

On note que les calculs sont faciles mais les résultats assez imprévisibles : justement, c'est le but que poursuivent nos deux cryptographes. La fonction $y = ax$ n'est pas monotone. L'exponentielle est définie :

$$\begin{aligned} 5^3 \bmod 11 &= 125 \bmod 11 \\ &= 4 \end{aligned}$$

et si n est premier elle a les mêmes propriétés que dans $\mathbb{Z}$:

$$(a^x)^y = (a^y)^x = a^{x.y}$$

Mise en œuvre de l'algorithme Diffie-Hellman

Voici maintenant le protocole d'échange de clés de Diffie-Hellman[1] [40], illustré par un exemple avec de petits nombres pour pouvoir faire les calculs à la main. Martin Hellman en a eu l'inspiration une nuit, mais il est le résultat de leur travail commun, auquel d'ailleurs il faut adjoindre Ralph Merkle. Le protocole repose sur une fonction de la forme $K = W^X \bmod P$, avec P premier et $W < P$. Une telle fonction est très facile à calculer, mais la connaissance de K ne permet pas d'en déduire facilement X. Cette fonction est publique, ainsi que les valeurs de W et P. Prenons $W = 7$ et $P = 11$, par exemple.

> **POUR ALLER PLUS LOIN**
>
> Le lecteur attentif remarquera que beaucoup d'auteurs utilisent cet exemple numérique. S'il se donne la peine de quelques essais personnels il constatera qu'il y a une bonne raison à cela : les autres valeurs numériques suffisamment petites donnent des résultats corrects mais peu pédagogiques du fait de coïncidences fâcheuses.

1. Aïcha choisit un nombre qui restera son secret, disons $A = 3$.
2. Boris choisit un nombre qui restera son secret, disons $B = 6$.
3. Aïcha et Boris veulent échanger la clé secrète, qui est en fait $S = W^{B.A} \bmod P$, mais ils ne la connaissent pas encore, puisque chacun ne connaît que A ou B, mais pas les deux.
4. Aïcha applique à A la fonction à sens unique, soit α le résultat :

$$\begin{aligned}\alpha &= W^A \bmod P \\ &= 7^3 \bmod 11 \\ &= 343 \bmod 11 \\ &= 2\end{aligned}$$

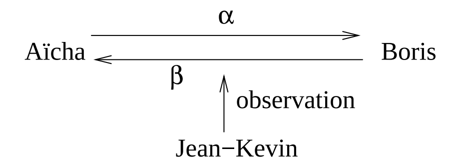
5. Boris applique à B la fonction à sens unique, soit β le résultat :

$$\begin{aligned}\beta &= W^B \bmod P \\ &= 7^6 \bmod 11 \\ &= 117\,649 \bmod 11 \\ &= 4\end{aligned}$$

[1] http://citeseer.ist.psu.edu/340126.html

6. Aïcha envoie α à Boris, et Boris lui envoie β, comme représenté par la figure 4.2. α et β ne sont pas la clé, ils peuvent être connus de la terre entière sans que le secret d'Aïcha et de Boris soit divulgué.

Figure 4.2
Échange de clés selon Diffie et Hellman



7. Aïcha a reçu β et calcule $\beta^A \bmod P$ (qui est, soit dit en passant, $(W^B)^A \bmod P$, soit $7^{B.A} \bmod 11$, mais elle ne connaît pas B) :

$$\begin{aligned} \beta^A \bmod P &= 4^3 \bmod 11 \\ &= 64 \bmod 11 \\ &= 9 \end{aligned}$$

8. Boris a reçu α et calcule $\alpha^B \bmod P$ (qui est, soit dit en passant, $(W^A)^B \bmod P$, soit $7^{A.B} \bmod 11$, mais il ne connaît pas A) :

$$\begin{aligned} \alpha^B \bmod P &= 2^6 \bmod 11 \\ &= 64 \bmod 11 \\ &= 9 \end{aligned}$$

Aïcha et Boris obtiennent à la fin de leurs calculs respectifs le même nombre 9 qui n'a jamais été exposé à la vue des indiscrets : c'est la clé S ! N'est-ce pas miraculeux ? Ils ont simplement échangé l'information nécessaire pour calculer la clé, sans divulguer celle-ci.

Supposons que Jean-Kevin veuille épier les conversations d'Aïcha avec Boris : il pourra intercepter l'échange des messages non chiffrés α et β, à partir desquels il veut calculer $S = \alpha^B \bmod P$. Il ignore S et B. L'équation à résoudre pour calculer B consiste à calculer la fonction réciproque de la fonction à sens unique :

$$W^B = \beta \bmod P$$

Si nous étions dans le monde des nombres réels la solution serait triviale :

$$B = \frac{\log \beta}{\log W}$$

Mais, dans le monde des classes d'équivalence modulo n, ce problème dit du *logarithme discret* n'a pas de solution simple, on ne connaît pas d'algorithme rapide pour le calculer. C'est un sujet de recherche. Le « front » est aujourd'hui à des valeurs de P qui sont des nombres de 450 chiffres binaires. L'algorithme est sûr si P a 512 chiffres binaires.

L'algorithme de Diffie-Hellman est sans doute une découverte majeure, totalement contraire à l'intuition. Il procure à deux acteurs d'un cryptosystème le moyen d'échanger une clé sans la faire circuler sur le réseau. Mais il restait à faire une découverte encore plus stupéfiante, inspirée d'ailleurs par celle que nous venons de décrire : un cryptosystème fondé sur des paires de clés dont l'un des deux éléments, la clé de chiffrement, est publiée dans des annuaires publics !

Le chiffrement asymétrique à clé publique

La méthode de Diffie et Hellman permet l'échange de clés, mais elle impose une concertation préalable entre les acteurs. Parfois ce n'est pas pratique : si Aïcha veut envoyer à Boris un courrier électronique chiffré pendant qu'il est en vacances, elle sera obligée d'attendre son retour pour établir la clé avec lui.

Whitfield Diffie avait eu une autre idée, pour laquelle il n'avait pas trouvé de solution mathématique appropriée : un système où l'on utiliserait une clé pour chiffrer et une autre pour déchiffrer. Ainsi, Boris proposerait à Aïcha une clé de chiffrement, avec laquelle elle coderait le message, et Boris le décoderait avec une clé différente, la clé de déchiffrement. La clé de chiffrement ne permet que de chiffrer, même Aïcha serait incapable de déchiffrer son propre message avec cette clé, seul Boris le peut avec sa clé de déchiffrement. Comme la clé de chiffrement ne fonctionne que dans un sens, elle permet de créer des secrets mais pas d'en dévoiler, et elle peut donc être publique, inscrite dans un annuaire ou sur un site Web. Quiconque veut envoyer un message chiffré à Boris peut la prendre et l'utiliser.

Il faut seulement être sûr que personne ne pourra calculer la clé de déchiffrement à partir de la clé de chiffrement. Et là l'intuition mathématique est décisive.

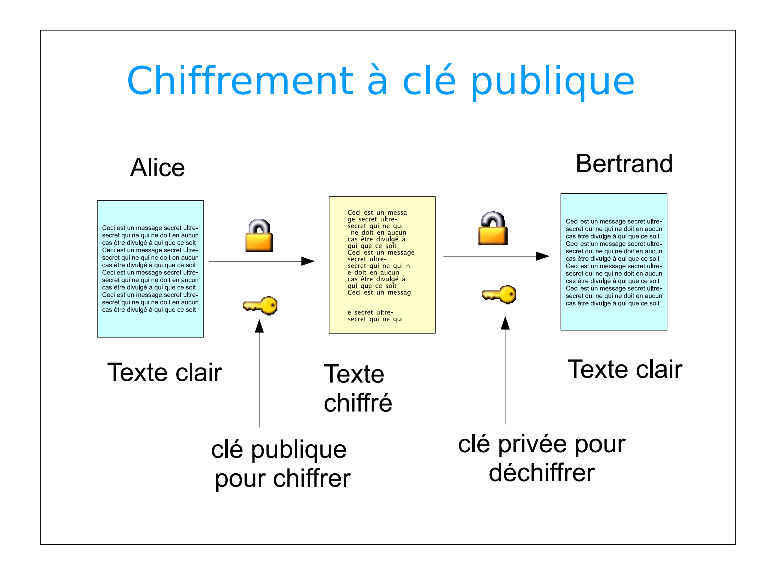


Figure 4.3
Principe du chiffrement asymétrique

Si l'idée du chiffrement asymétrique à clés publiques revient à Diffie et Hellman (sans oublier les précurseurs britanniques tenus au secret), la réalisation de cette idée revient à Rivest, Shamir et Adleman, : ils ont trouvé une solution mathématique permettant sa mise en œuvre et nommé cette solution [91] de leurs initiales : RSA.

Une personne désireuse de communiquer selon cette méthode doit procéder ainsi :

1. Prendre deux nombres premiers p et q. En cryptographie réelle on choisira de très grands nombres, de 150 chiffres décimaux chacun. Nous allons donner un exemple avec $p = 3$ et $q = 11$.
2. Calculer $n = pq$, soit dans notre exemple $n = 33$.
3. Calculer $z = (p - 1)(q - 1)$. (Ce nombre est la valeur de la fonction $\phi(n)$, dite fonction phi d'Euler, et on notera qu'elle donne la taille du groupe multiplicatif modulo n, $\mathbb{Z}_n^*$). Dans notre exemple $z = 20$.
4. Prendre un petit entier e, impair et premier avec z, soit $e = 7$. Dans la pratique, e sera toujours petit devant n[2].

[2] e est dit *premier avec* z s'ils n'ont aucun diviseur commun autre que 1 et -1, et on dira que e est petit devant n s'il est beaucoup plus petit que n.

5. Calculer l'inverse de $e \pmod z$, c'est-à-dire d tel que $e.d = 1 \bmod z$. Les théorèmes de l'arithmétique modulaire nous assurent que, dans notre cas, d existe et est unique. Dans notre exemple $d = 3$.
6. La paire $P = (e, n)$ est la clé publique.
7. Le triplet $S = (d, p, q)$ est la clé privée.

Voyons ce que donne notre exemple. La clé publique de Boris est donc (7, 33). Aïcha veut lui envoyer un message M, disons le nombre 19. Elle se procure la clé publique de Boris sur son site Web et elle procède au chiffrement de son message M pour obtenir le chiffré C comme ceci :

$$C = P(M) = M^e \bmod n$$
$$C = P(19) = 19^7 \bmod 33 = 13$$

Pour obtenir le texte clair T, Boris décode avec sa clé secrète ainsi :

$$T = S(C) = C^d \bmod n$$
$$T = S(13) = 13^3 \bmod 33 = 19$$

Miraculeux, non ? En fait c'est très logique :

$$\begin{aligned} S(C) &= C^d \bmod n \\ &= (M^e)^d \bmod n \\ &= M^{e.d} \bmod n \\ &= M \bmod n \end{aligned}$$

Le dernier résultat, $M^{e.d} = M \pmod n$ découle du fait que e et d sont inverses modulo n, il se démontre grâce au petit théorème de Fermat.

À quel type d'attaque est exposé RSA ? Un espion (Jean-Kevin par exemple) pourra obtenir la clé publique de Boris $P = (e, n)$, qui a servi à chiffrer M, ainsi que le message chiffré, C. Pour trouver M l'équation à résoudre est :

$$C = M^e \bmod n$$

n, C et e étant connus. Encore une fois, dans le monde des réels, la solution est triviale : $M = \sqrt[e]{C}$. Mais dans le monde modulaire la solution est $M = \sqrt[e]{C} \bmod n$, et il n'y a pas d'algorithme rapide connu pour la calculer, même pour de petites valeurs de e. Ainsi, trouver la racine cubique modulo n d'un nombre y est un problème qui n'est toujours pas résolu.

En fait la seule attaque possible (outre la recherche de failles de réalisation du logiciel) consisterait à trouver p et q par recherche des facteurs de n, ce que l'on appelle la factorisation du nombre n. La factorisation permettrait de calculer $z = \phi(n) = (p-1)(q-1)$. Le nombre secret d est tel que $e.d \equiv 1 \bmod z$. d est un nombre du même ordre de grandeur que z, soit un nombre de mille chiffres binaires. Ce calcul serait réalisable, mais l'obstacle est que la factorisation n'est pas un problème résolu, et qu'il est donc impossible, dans le cas général, de calculer p, q et z.

Les réalisations industrielles ont longtemps utilisé, et utilisent parfois encore $e = 3$. De nos jours $e = 2^{16} + 1 = 65\,537$ est populaire. Avec un tel choix, d est du même ordre de grandeur que n, soit $d \approx 2^{1024}$. L'élévation à une puissance de cet ordre peut être réalisée efficacement par des algorithmes de type « élévation au carré et multiplication » *(square and multiply)*, qui prennent moins d'une seconde dans une carte à puce.

POUR EN SAVOIR PLUS

Le lecteur trouvera des explications mathématiques supplémentaires dans l'ouvrage de Cormen, Leiserson et Rivest (le R de RSA) [35] ou dans celui de Menezes, van Oorschot et Vanstone [79], ou encore, de façon plus abordable, dans ceux de Gilles Dubertret [43] ou d'Albert Ducrocq et André Warusfel [44]. Au demeurant, il est stupéfiant de constater que les découvertes prodigieuses de Diffie, Hellman, Merkle, Rivest, Shamir et Adleman reposent sur des bases mathématiques déjà entièrement établies par Leonhard Euler (1707–1783), sinon par Pierre de Fermat (1601–1665), et que, hormis Donald Knuth dans les années 1960, personne n'y avait pensé avant. Et si personne n'y avait pensé, c'est qu'avant l'invention de l'informatique moderne les méthodes cryptographiques dont nous venons de donner un bref exposé étaient non seulement irréalisables, mais *impensables*.

Évaluer la robustesse d'un cryptosystème

Comme nous l'avons vu, les systèmes de chiffrement symétriques et asymétriques reposent sur des méthodes mathématiques complètement différentes. Ces deux familles de systèmes sont d'un usage complémentaire, ils sont utilisés conjointement dans les réalisations techniques que nous employons quotidiennement. Il convient d'avoir une conscience claire du fait que la confiance que l'on peut placer en eux, ou pas, ou en d'autres termes leur *robustesse*, repose sur des hypothèses de nature radicalement différentes dans les deux cas.

Robustesse du chiffrement symétrique

La robustesse d'un système de chiffrement symétrique repose sur l'impossibilité de deviner la clé utilisée : ce qui découle d'une précaution et de trois qualités :

- la précaution est que les utilisateurs doivent éviter de divulguer la clé et la stocker sur un support convenablement protégé, cela semble évident mais souvent cette condition n'est pas vérifiée ;
- l'espace des clés doit être vaste, pour parer aux *attaques par force brute* qui consistent à tenter le déchiffrement avec toutes les clés possibles successivement ; autrement dit, la clé doit comporter beaucoup de chiffres ;
- l'algorithme doit être lui-même robuste, c'est-à-dire tel que l'examen du message chiffré ne doive pas révéler d'indices de nature à aider le déchiffrement, soit par la découverte de la clé, soit par l'élucidation directe du message ;
- enfin la réalisation du logiciel doit être correcte, et c'est généralement là que gisent les failles ; les algorithmes robustes sont complexes et subtils, une programmation maladroite peut réduire de façon spectaculaire la taille réelle de l'espace des clés ; or c'est de la taille de l'espace des clés que découle l'entropie du cryptosystème, qui est la mesure mathématique de sa quantité d'incertitude ou d'aléa.

Pour pouvoir décerner un certificat de robustesse à un système de chiffrement symétrique, on doit démontrer que l'attaque par force brute est l'attaque optimale. Ensuite, cela acquis, la guerre entre cryptographes et cryptanalystes se résume à une question de longueur de clé et de puissance de calcul.

En pratique, les assaillants réels cherchent (et trouvent) des failles de réalisation dans les systèmes réels, cependant que les assaillants du monde de la recherche

constituent des grilles de calcul intercontinentales pour casser les algorithmes, ce qui est utile aux progrès de la science, mais peu utilisable par des cyber-criminels.

Robustesse du chiffrement asymétrique

La robustesse des cryptosystèmes à clés publiques repose sur deux piliers :

- La confidentialité de la clé privée de celui qui l'utilise ; en effet la divulgation de cette clé privée réduit à néant la protection offerte par le système.
- Les résultats de la théorie des nombres, ou plutôt l'absence de tels résultats, qui nous dit que la factorisation de très grands nombres est un problème difficile, ainsi d'ailleurs que le problème du logarithme discret (ici, le terme *difficile* doit être entendu comme *insoluble en pratique*). C'est-à-dire que tous ces systèmes sont à la merci d'un progrès inattendu de la théorie mathématique, qui viendrait par exemple offrir aux cryptanalystes un nouvel algorithme de factorisation rapide.

Comme pour le chiffrement symétrique, les nombres choisis comme bases du système doivent être suffisamment grands pour décourager les attaques par force brute, et la réalisation des programmes doit être correcte.

Robustesse de l'utilisateur de cryptosystème

Ainsi les deux types de cryptosystèmes obéissent à des critères de robustesse assez différents, mais la confiance qu'il est possible de leur accorder repose aussi sur un facteur qu'ils ont en commun : la responsabilité de l'utilisateur. Nous avons déjà évoqué à la page 14 la thèse de Marcus J. Ranum au sujet de l'éducation des utilisateurs, et nous y reviendrons à la page 224 : si la sûreté de votre système repose sur l'éducation des utilisateurs, alors il convient d'être inquiet. Un autre cryptologue célèbre, Bruce Schneier, a écrit un article[3] où il soupèse la confiance que l'on peut placer dans un autre type de systèmes de sécurité dont nous parlerons plus loin (page 176), mais son analyse rejoint celle de Ranum sur ce point : les systèmes de sécurité reposent sur des comportements humains dont il est imprudent de penser qu'ils seront adoptés toujours et en toutes circonstances :

[3] http://www.schneier.com/paper-pki.html

- Qui peut garantir que sa clé privée est inaccessible ?
- Est-il possible pratiquement de vérifier l'authenticité des certificats électroniques de tous les sites que nous visitons ?
- Et si nous nous avisions de le faire, les serveurs qui détiennent les listes de révocations de certificats et les réseaux qui leur donnent accès ne seraient-ils pas irrémédiablement saturés ?
- Nous pouvons être sûrs de l'identité d'un porteur de certificat ou de clé publique que nous avons rencontré en personne et qui nous a montré une pièce d'identité et le condensat de sa clé, mais faire confiance à un certificat parce qu'il est signé par une lointaine autorité de certification n'est pas si facile : le nom et le prénom d'une personne peuvent l'identifier de façon sûre au sein d'une petite population, mais ce n'est plus vrai à l'échelle d'un pays ou du monde. Et son identité a pu être usurpée. C'est tout le problème de l'attaque par interposition (*Man in the middle*).

Il n'est donc guère raisonnable d'espérer que les utilisateurs soient en mesure de garantir que ces conditions seront réunies en tout lieu et à chaque instant. Nous envisagerons des moyens de progresser vers ce but, sinon de l'atteindre, au chapitre 7, page 175.

Comment générer une paire de clés RSA

Pour satisfaire une légitime curiosité, voici la commande à utiliser avec le logiciel libre *OpenSSL* pour générer un certificat électronique x509 et la clé privée associée, ainsi que le résultat de cette commande :

```
# openssl req -x509 -newkey rsa:1024 -days 365 -keyout marti.pem -out marti.pem
-----BEGIN RSA PRIVATE KEY-----
Proc-Type: 4,ENCRYPTED
DEK-Info: DES-EDE3-CBC,4AB14B73133CE78D
wNvqLBqSwq7BsONOZfCrXyZYKQ8OIAue2n/c2ISuIpLXXEnJ7Ku85LOY9uj/Dk9e
Cg6rv3j73N8ZBJYA/86xrq/G37jn0U2wRtsQWtoFdSxldzDrhKoBv64hhaiYbGix
c4BpsTmYt+9lRPgi2KcMN2IwKcEBSS1Ifk1mls5g5v6KvrqpkTYLRlnkPKzps3B2
uldjoXea4E03L1jxzjq+t6R6+E+fcBaHxEMTq4IBjSLnVp+XRaer2WKaJguXed6q
Yt/MRLHr7VtO0J6IHC9qRyCr9K80ykY1CTpN5c1eam+1uYEErYYJUwCBvrU8F12l
1xBZ3z2HoocztxNWtdatZgABMIOSd1uHa9ERg/Cr6KGHpbNO2iiT20G3UmxfF06S
vLXeRjqnNmAbbnK3aEyAS1zYJylxU/ttyPdObled8nlA0CigZ3LiqKOwz02NgDIh
bWpqQAyiBJHw2VDuBLP3Ks6v0v749IKXPeopVLTjSjuvMcNyP1oaI2CjYrcJbt3p
8Bt6JxuIlJqE1iTtARSBvDVpKmWjRfVtYnU5SXpJwqYz70hsx1eR/evT3M9sDkUl
t7KAvOSu7Pe8sYl9oD/rQ52XkpsEqhYv5o46gMhC3hJILnqxBi3aPnAuuaBs4Ugk
0vbJ7sps6QBQ2vdHn/bJTq+SqAeXo8PssMRdsW/wAih3dEAxEM4QYiFJCCpG4/eX
yD8itPrrozm3e/geBIfZck+aZLa1yWvE6vlKKhn8OugHC+M0tGKzOg5EYT9Utk5R
2sgYVnoE3mKmqYfKVSafa3F/WFTepCqaIC/JKmUz2bc28slqHdzlNA==
-----END RSA PRIVATE KEY-----
```

5

Sécurité du système d'exploitation et des programmes

Un modèle de protection : Multics

Dans le domaine de la protection, l'approche mise en œuvre par le système Multics dès les années 1960 fait encore aujourd'hui figure de référence. Nous allons la décrire.

Multics est né en 1964 au MIT (*Massachusetts Institute of Technology*) dans le cadre d'un projet de recherche nommé MAC, sous la direction de Fernando Corbató. L'objectif du projet était la réalisation d'un grand système informatique capable de fournir des services interactifs en temps partagé à un millier d'utilisateurs simultanés. Multics comportait beaucoup d'innovations de grande portée : le langage de commande pour piloter le fonctionnement de la machine était un langage de programmation, le même que celui dont disposait l'utilisateur pour interagir avec le système, le *shell* inventé à cette occasion.

Le système d'exploitation était écrit en langage évolué (en l'occurrence PL/1), voie ouverte par les systèmes Burroughs écrits en Algol, mais encore peu fréquentée. Les concepts de mémoire centrale pour les données volatiles et de fichiers pour les données persistantes étaient fondus en un concept unique de mémoire virtuelle segmentée, certains segments étant dotés de la qualité de persistance.

De même que les auteurs de Multics avaient accompli une percée conceptuelle considérable et qui reste aujourd'hui à poursuivre en réunissant les structures de données en mémoire et les fichiers persistants sur disque en un concept unique de segment, ils ont aussi imaginé pour la protection une approche et des concepts originaux et puissants que les systèmes d'aujourd'hui redécouvrent lentement.

Abolir les fichiers !

L'unification conceptuelle des données en mémoire vive et des données persistantes (fichiers) fut un progrès parce qu'elle abolit une distinction arbitraire (mémoire–fichier) dont les raisons techniques sont aujourd'hui en grande partie périmées grâce aux systèmes à mémoire virtuelle et à la capacité accrue des mémoires de toutes sortes. Elle reste à poursuivre parce que les systèmes actuels reposent encore sur l'ancien modèle de mémoire et sur la notion inélégante de fichier.

Si Multics n'a guère connu le succès, sa postérité est innombrable, parce qu'Unix (et par conséquent Linux) en sont les descendants directs. À la fin des années 1960, l'échec de Multics aux *Bell Labs* était patent. L'équipe qui allait y concevoir Unix, autour de Ken Thompson et Dennis Ritchie, comprit que Multics ne serait pas utilisable pour un travail réel dans un délai raisonnable. Le groupe de D. Ritchie, K. Thompson, M. D. McIlroy et Joseph F. Ossanna souhaitait conserver l'environnement de travail luxueux que Multics leur procurait à un coût d'autant plus exorbitant qu'ils en étaient les derniers utilisateurs. Pour ce faire ils allaient développer leur propre système sur un petit ordinateur bon marché et un peu inutilisé récupéré dans un couloir, un PDP 7 de Digital Equipment. Unix était sinon né, du moins conçu, mais il abandonnait certains des concepts novateurs de Multics, notamment l'unification mémoire vive – mémoire persistante et les dispositifs de protection, trop coûteux en mémoire et en temps de processeur pour les ordinateurs de l'époque.

Les dispositifs de protection de Multics

Nous décrirons les dispositifs et procédures mis en œuvre dans Multics pour assurer la protection des objets car, bien qu'anciens, ils restent à ce jour de l'an 2006 une réalisation de référence. Cette description doit beaucoup à celles de l'ouvrage collectif de CROCUS [37], *Systèmes d'exploitation des ordinateurs* et du livre de Silberschatz et ses collègues [102] *Principes appliqués des systèmes d'exploitation*.

La protection sous Multics repose sur une structure dite « en anneaux ». Chaque processus s'exécute dans un anneau, chaque anneau correspond à un niveau de privilèges. Multics offre huit anneaux numérotés de 0 à 7, l'anneau 0 procure les privilèges les plus élevés, l'anneau 7 les moins élevés. L'anneau du processus courant figure dans le mot d'état de programme (PSW), zone de mémoire qui contient des informations essentielles sur le traitement en cours, notamment l'adresse de la prochaine instruction à effectuer.

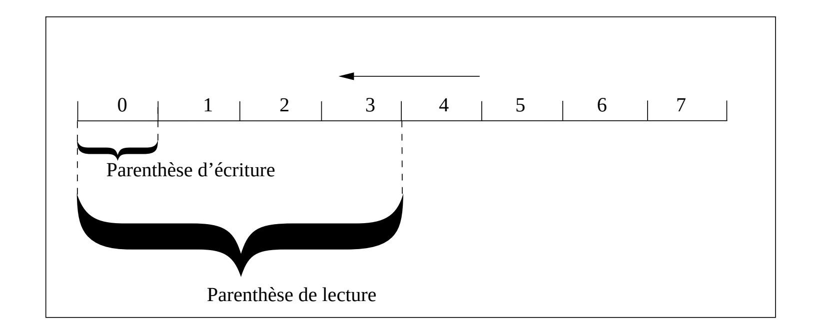
Chaque segment (de mémoire volatile ou persistante), pour chaque type d'accès (lecture, écriture, exécution si le segment contient un programme ou un répertoire), appartient à un anneau. Si un processus s'exécute dans un anneau de valeur inférieure ou égale à l'anneau d'exécution d'un segment, par exemple, il peut exécuter le programme contenu dans ce segment, sinon non. Le schéma 5.1 page suivante donne un exemple de ce mécanisme : il représente la protection d'un segment accessible en lecture à des processus qui s'exécutent dans les anneaux 0 à 3, mais dont la modification par une écriture est réservée aux processus de l'anneau 0.

À tout moment un processus peut changer d'anneau (sous le contrôle du système d'exploitation qui évidemment vérifie que ce processus dispose des accréditations nécessaires) et ainsi acquérir de façon temporaire ou définitive des privilèges supérieurs qui lui ouvriront l'accès à de nouveaux segments.

Protection des systèmes contemporains

Finalement il apparaît que les plus fidèles disciples de l'équipe Multics furent les ingénieurs d'Intel. Depuis le modèle 80286 jusqu'à l'actuel Itanium les processeurs de la ligne principale d'Intel disposent d'une gestion de mémoire virtuelle à adressage segmenté et d'un système de protection à quatre anneaux, typiquement destinés respectivement au noyau du système pour l'anneau 0, aux fonctions auxi-

Figure 5.1
Protection en anneaux sous *Multics*



liaires du système pour les anneaux 1 et 2, et aux programmes en mode « utilisateur » pour l'anneau 3. Ces possibilités des processeurs Intel ne sont guère utilisées par les systèmes d'exploitation, que ce soient ceux de Microsoft ou les Unix libres FreeBSD, OpenBSD, NetBSD ou Linux ; aucun ne tire parti de ce dispositif pour unifier les gestions de la mémoire virtuelle et de la mémoire persistante (le système de fichiers) : les premiers sont contraints à la compatibilité avec leurs ancêtres... et les Unix aussi.

Les systèmes conventionnels comme Unix possèdent un système d'anneaux dégradé à seulement deux anneaux (le mode superviseur et le mode utilisateur) et un système de listes d'accès dégradé avec pour chaque fichier des droits d'accès en lecture, en écriture et en exécution pour trois ensembles d'utilisateurs : le propriétaire du fichier, les membres de son groupe, tous les autres utilisateurs. Linux utilise l'anneau 0 comme mode noyau et l'anneau 3 comme mode utilisateur et c'est tout. Ces systèmes plus rudimentaires ont (avaient ?) l'avantage d'être moins lourds.

Débordements de tampon

Si vous consultez un site de publication de listes de vulnérabilités, par exemple celui de *Security Focus*[1], vous verrez apparaître avec une fréquence étonnante la phrase « *A buffer overflow allows remote attackers to execute arbitrary code...* » (« Un

[1] Cf. `http://www.securityfocus.org/`

débordement de tampon permet à un agresseur à distance d'exécuter un code arbitraire... »), c'est-à-dire que cet agresseur est en mesure de prendre le contrôle de l'ordinateur affecté et d'en faire n'importe quoi, ce qui représente la gravité maximale pour un incident de sécurité.

De quoi s'agit-il ? D'une maladie fréquente qui fait l'objet d'une entrée dans Wikipédia[2] et de nombreux articles détaillés, celui-ci[3] parmi beaucoup d'autres. Nous commencerons par en exposer un cas particulier très significatif, qui a le mérite de bien faire comprendre le principe du mécanisme, puis le cas général.

Mais il convient d'abord de préciser ce que l'on entend par *tampon (buffer)* : ce terme désigne en général une zone de mémoire utilisée par le système d'exploitation pour stocker temporairement des données en provenance du (ou en partance vers le) monde extérieur. Plus généralement il s'agira ici d'une zone de mémoire utilisée par un programme pour y manipuler un texte, au sens le plus général.

Attaques par débordement sur la pile

La revue *Communications of the Association for Computer Machinery* a récemment publié un article [71] qui décrit en détail un cas particulier d'attaque par débordement de tampon, celui qui survient sur la *pile* du programme.

Le principe en est le suivant : un programme en cours d'exécution note dans un coin (comme sur un post-it) l'adresse à laquelle revenir quand il aura fini son travail (la notion d'adresse est précisée plus bas). L'attaquant cherchera à modifier cette adresse de retour pour la remplacer par celle d'un programme malveillant installé par ses soins.

Le « coin » où cette adresse de retour est notée se situe dans une zone nommée pile d'exécution du programme. Une pile est une structure de données, une façon d'organiser un ensemble de données, telle que la dernière donnée introduite sera la première à être obtenue, on parle d'organisation LIFO, comme *last in, first out*, comme pour une pile d'assiettes, où la première à prendre sera celle du dessus, qui a été déposée la dernière. Pour extraire les données de la pile, on utilise un *pointeur* ; un pointeur est une donnée dont la valeur est un moyen d'accès à la valeur d'une autre donnée ; par exemple la valeur du pointeur peut être le numéro

[2] Cf. `http://fr.wikipedia.org/wiki/Buffer_overflow`
[3] Cf. `http://c2.com/cgi/wiki?CeeLanguageAndBufferOverflows`

de la case mémoire où se trouve la valeur de la donnée pointée par lui, autrement dit ce que l'on appelle habituellement son *adresse*. Le *pointeur de pile (stack pointer)* est une donnée qui contient l'adresse du dernier élément empilé, ou en d'autres termes l'adresse du sommet de la pile.

Un programme est généralement constitué de plusieurs sous-programmes ; à chaque sous-programme correspondent des données locales : les valeurs des paramètres transmis lors de l'appel du sous-programme, des variables locales dont la durée de vie est limitée à la période d'activité de ce sous-programme, et surtout l'adresse de retour vers le programme appelant, c'est-à-dire l'adresse de l'instruction qui devra être exécutée à la fin du sous-programme. L'ensemble de ces données locales constitue le *bloc d'activation (activation record)*[4] de *cet* appel au sous-programme, aussi appelé *cadre de pile (stack frame)*.

La pile d'exécution d'un programme est une pile de blocs d'activation. Lors d'un appel à un sous-programme, le bloc d'activation (ou cadre de pile) correspondant est créé et stocké sur la pile du programme. Lorsque le sous-programme se termine, ce cadre de pile est détruit et le pointeur de pile pointe vers le cadre précédent, qui est celui du programme appelant.

On observe qu'une telle organisation en pile autorise que soient présents en mémoire à un instant donné les blocs d'activation de plusieurs appels emboîtés au même sous-programme, ce qui est indispensable pour les programmes récursifs, qui s'appellent eux-mêmes ; ainsi chaque instance du sous-programme possède son propre bloc d'activation, avec ses variables locales et ses arguments, sans interférence avec les autres instances. La figure 5.2 représente la pile d'un processus Unix ; on notera qu'elle croît à l'envers, son sommet est vers le bas (vers les adresses de plus faibles valeurs) ; le *tas* est une autre région de la mémoire du programme, où sont allouées les zones nécessaires à des données de plus grande taille, tels les tableaux.

Un débordement de tampon sur la pile consistera à altérer de façon fautive mais soigneusement calculée, dans un programme légitime, une variable locale de type chaîne de caractères, de façon que l'adresse de retour soit écrasée par l'adresse du code malicieux placé là par le pirate ; ainsi ce sera ce code qui s'exécutera à la fin de l'exécution du sous-programme.

[4] Les auteurs de langue française traduisent souvent *activation record* par *enregistrement d'activation*, qui me semble moins approprié que *bloc d'activation*.

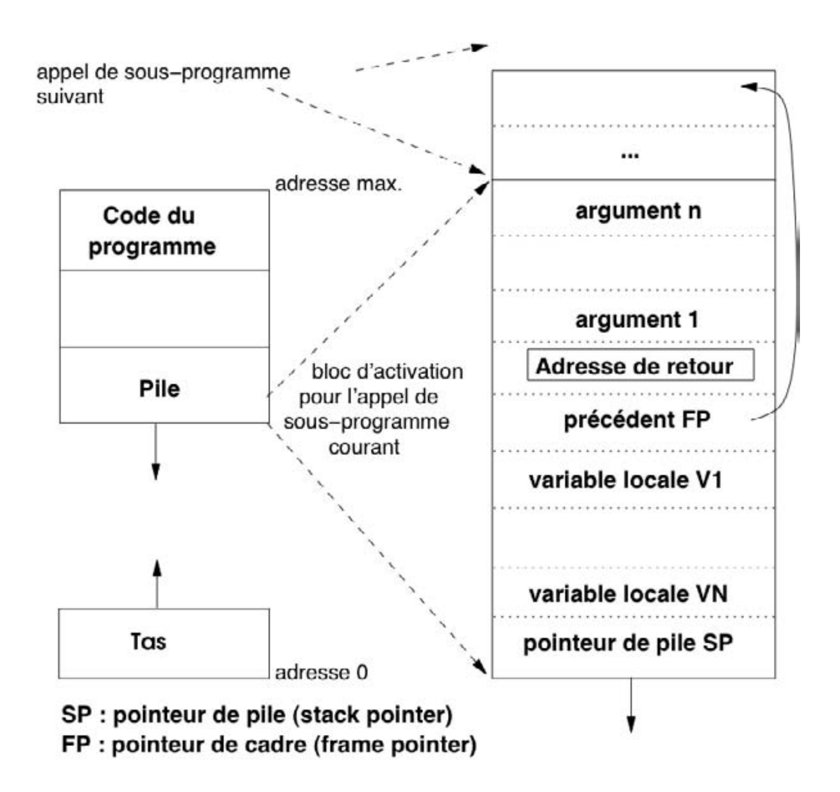


Figure 5.2
La pile d'un processus Unix

ALTERNATIVE **Faire croître la pile vers le haut ?**

Christian Queinnec me fait observer la chose suivante : cette possibilité d'écraser l'adresse de retour d'un programme par un débordement de tampon résulte uniquement du choix des concepteurs d'Unix (imités par de nombreux suiveurs) de faire croître la pile vers le bas ; si les blocs d'activation s'empilaient dans l'autre sens, les débordements de tampons écraseraient sans doute des choses, mais pas l'adresse de retour. Il y a bien sûr des raisons à ce choix de conception : faire croître la pile et le tas en sens inverse simplifie l'utilisation de la mémoire, procéder autrement poserait d'autres problèmes, mais devant l'abondance des failles qui reposent sur ce dispositif, on devrait au moins se poser la question.

L'article des CACM cite quelques attaques réussies par débordement de tampon sur la pile, et énumère plusieurs remèdes de nature à les prévenir :

1. les développeurs du projet *OpenBSD* passent en revue le code de l'ensemble de leur système pour y repérer les séquences d'instructions affectées par une telle vulnérabilité, et ajouter des contrôles de nature à vérifier la longueur des chaînes de caractères concernées ; ils ont créé, pour les aider dans ce travail de bénédictin, des logiciels qui automatisent une partie des opérations ;
2. il existe des logiciels ou des bibliothèques de sous-programmes qui instrumentent le compilateur de façon qu'il insère automatiquement des contrôles adéquats dans le programme compilé :
   - *StackGuard* place dans la pile un marqueur spécial, et détecte grâce à lui toute altération anormale de l'adresse de retour,
   - *StackShield* modifie le comportement du compilateur `gcc` de façon à ce qu'il insère dans le programme compilé des séquences d'instructions destinées à maintenir une copie de secours de la pile des adresses de retour, et à détecter ainsi toute altération anormale de la pile du processus,
   - RAD est une modification du compilateur `gcc` qui insère au début et à la fin des appels de fonctions des instructions qui recopient la pile des adresses de retour, un peu comme *StackShield* ;
3. il existe également un projet baptisé *SmashGuard* qui se propose d'implanter les opérations destinées à protéger la pile contre les débordements de tampons dans le matériel, par la modification des instructions d'appel et de retour de fonction, ce qui éviterait d'une part de modifier ou de recompiler d'innombrables logiciels, d'autre part de détériorer les performances des systèmes en voulant les rendre plus sûrs.

CULTURE gcc

gcc est le compilateur de base du projet GNU. Un compilateur est un programme qui traduit le texte d'un programme vers un langage de plus bas niveau, souvent le langage machine de l'ordinateur sur lequel on souhaite que le programme s'exécute. Initialement conçu pour le langage C, gcc sert désormais à traduire d'autres langages, tels que C++, Ada ou Java. C'est un logiciel libre.

Débordement de tampon : exposé du cas général

Pour citer Wikipédia, « le principe [du débordement de tampon] est de profiter de l'accès à certaines variables du programme, souvent par le biais de fonctions telles `scanf()` (analyse de chaîne de caractères) ou `strcpy()` (copie de chaîne de caractères) en langage C, qui ne contrôlent pas la taille de la chaîne à enregistrer dans un tampon, afin d'écraser la mémoire du processeur jusqu'à l'adresse de retour de la fonction en cours d'exécution. On peut ainsi choisir quelles seront les prochaines instructions exécutées par le processeur. Le code introduit est généralement exécuté avec les droits du programme détourné. »

Comme nous l'expliquent Cunningham & Cunningham [38], les langages C et C++ représentent les chaînes de caractères au moyen de tampons de caractères (`char *`), qui ne sont que des zones de mémoire à partir d'une adresse de début, dépourvues de taille propre. La fin de la zone est simplement indiquée par le caractère ASCII 0, que le programme peut écraser à sa guise pour allonger la taille attribuée à la chaîne considérée. L'utilisateur malveillant d'un programme écrit en C peut fournir, en réponse à une question du programme, une chaîne de caractères plus longue que ce que le programmeur a prévu, et ainsi écrire subrepticement dans des zones mémoires réservées à d'autres usages, ce qui va altérer le comportement du programme. Par exemple, il sera possible ainsi d'insérer, à des emplacements bien choisis, du code exécutable, et de le faire exécuter par le programme : c'est l'*exploitation* d'un débordement de tampon.

Que le programme soit vulnérable à ce genre d'exploitation n'est bien sûr pas une fatalité : il est possible d'écrire des programmes qui vérifient que la longueur de la chaîne de caractères saisie par l'utilisateur n'excède pas la taille prévue pour le tampon, mais c'est très laborieux, et l'expérience montre que, sauf à utiliser des méthodes systématiques, le programmeur en oublie toujours quelque part. Parmi les méthodes systématiques on peut citer l'usage de bibliothèques de fonctions de traitement de chaînes de caractères écrites conformes aux bonnes règles de sécurité ; de telles bibliothèques existent mais leur usage n'est pas très répandu parce qu'elles sont assez étrangères à la tradition de la programmation en C.

Débordement de tampon et langage C

Il est à noter que le problème du débordement de tampon est propre aux langages C et C++ : ainsi, il est à peu près impossible d'en déclencher un en Java,

en Scheme, en Perl ou en Python, parce que ces langages sont plus restrictifs et ne comportent pas de pointeurs qui contiennent des adresses de zones arbitraires en mémoire, modifiables par le programmeur. Même à supposer que le traducteur du langage ou la machine virtuelle chargée de l'exécuter soient eux-mêmes affectés d'erreurs, leur exploitation malveillante serait très difficile.

Cette possibilité qu'offre le langage C d'utiliser des adresses explicites qui désignent des emplacements arbitraires en mémoire a ses raisons d'être : C a été conçu à l'origine pour écrire un système d'exploitation (Unix en l'occurrence), et un auteur de système d'exploitation a besoin d'accéder à des zones de mémoire situées à des emplacements physiques arbitraires, ne serait-ce que... pour gérer la mémoire elle-même. Cette possibilité n'est d'aucune utilité réelle pour un serveur Web ou pour un logiciel d'affichage de documents, et elle devient au contraire une nuisance. Le malheur est que la popularité de C, accrue par des qualités extra-langagières telles que la disponibilité sans supplément de coût sur tout système Unix et la relative légèreté du compilateur, ont contribué à sa propagation auprès de communautés de développeurs d'applications, pour lesquels il n'était sans doute pas le mieux adapté.

Sécurité par analyse du code

Les lignes qui suivent concernent sans doute plus les auteurs de logiciels que les administrateurs de réseaux. Mais quel administrateur n'est pas à ses heures développeur ? Ne serait-ce que de logiciels d'administration, souvent écrits en *Perl*, langage dont il sera question ci-dessous.

Analyses statiques et méthodes formelles

L'*analyse statique de programme (static code analysis)* désigne un ensemble de techniques destinées à déterminer certaines propriétés d'un programme sans déclencher son exécution, par opposition aux méthodes de test. L'énoncé même de ce projet montre qu'il peut avoir des applications dans le domaine de la sécurité.

Une première famille de méthodes qui se rattachent à cette catégorie comporte des méthodes formelles :

- la *sémantique dénotationnelle* se propose de créer le modèle sémantique d'un système informatique en construisant des objets mathématiques qui expriment sa sémantique, ou en d'autres termes ce qu'il fait ;
- la *sémantique axiomatique* vise le même but en s'appuyant sur des formalismes empruntés à la logique ;
- la *sémantique opérationnelle* utilise aux mêmes fins les diagrammes d'état et l'interprétation symbolique.

Entrer dans le détail de ces méthodes nous entraînerait au-delà du champ de cet ouvrage, on pourra se reporter aux références indiquées par Wikipédia[5].

Méthode B

Devant la difficulté de mise en œuvre des méthodes formelles évoquées à la section précédente, Jean-Raymond Abrial a choisi d'aborder ce problème par une autre face : prouver la justesse et la sûreté du programme avant de l'écrire, et pour cela il a créé la méthode B [7],[15] au milieu des années 1980. Elle a été utilisée dans le cadre du projet de métro sans conducteur METEOR (Métro est-ouest rapide) réalisé par Matra (maintenant *Siemens Transportation System*) pour le compte de la Régie autonome des transports parisiens (RATP), pour construire de façon sûre les parties du logiciel qui jouent un rôle critique pour la sécurité des passagers. La partie du logiciel du métro METEOR réalisée grâce à l'Atelier B (l'outil informatique sous-jacent à la méthode [31] développé par la société ClearSy) comprend près de 100 000 lignes de code Ada générées automatiquement. Notons qu'auparavant B Abrial avait créé le langage de spécification Z : peut-être un clin d'œil de cinéphile aux amateurs des films de série B ou Z ?

Les premières démarches de preuve de programme tentaient d'appliquer des procédures de preuve à des programmes déjà construits. Il s'est assez vite révélé qu'un programme final était un objet beaucoup trop complexe pour être soumis d'un seul coup à une procédure de preuve, manuelle ou à plus forte raison automatique. L'idée de B est donc d'élaborer la preuve en même temps que le programme. Le langage de développement B permet de spécifier d'une part le programme proprement dit, d'autre part les propriétés dont on souhaite le voir doté.

[5] `http://en.wikipedia.org/wiki/Denotational_semantics`
`http://en.wikipedia.org/wiki/Hoare_logic`
`http://en.wikipedia.org/wiki/Operational_semantics`

On aura compris que B est un système de développement complet, qui comporte son propre langage de programmation, ce qui confirme en fait l'idée qu'une méthode de spécification est soit un langage de programmation, soit inutile, et que de toute façon la création d'un système informatique demande une vraie compétence en programmation. Il semble clair que les exigences de B en termes de délais et de qualification technique du personnel sont relativement élevées par rapport à du développement classique de logiciel non critique. Le tout est de ne pas se tromper dans la détermination de ce qui est critique et de ce qui ne l'est pas.

Lors du développement des 100 000 lignes de code Ada critique pour le logiciel du métro METEOR, l'atelier B a produit 30 000 obligations de preuve, dont plus de 90% ont été réalisées automatiquement. Les quelque 2 500 preuves qui ont résisté aux procédures automatiques ont nécessité plusieurs mois de travail humain, mais l'industriel a estimé que le bilan était largement positif grâce à l'économie engendrée par la suppression des tests de bas niveau. Après quelques années d'exploitation sans incident notable, la conclusion s'impose que la méthode B est efficace et sûre pour les développements critiques.

Pour être complet il faut également signaler les limites de la méthode B :

- les capacités de preuve sur des formules comportant des opérations arithmétiques sont limitées ;
- le développement formel de systèmes contenant des calculs numériques n'est actuellement pas possible avec la méthode B et les outils associés ; de tels calculs restent sous-spécifiés et les preuves de correction ne peuvent être données ;
- absence de vérification de propriétés temporelles due à la logique supportée (par l'atelier B) ;
- on ne peut pas décrire avec B les phénomènes concurrents, les fenêtres de temps et plus généralement le temps réel (codage des événements par des variables) ; les logiques temporelles sont plus adaptées pour spécifier le comportement dynamique.

Perl en mode souillé

Le langage *Perl* [5] propose une méthode de sécurité statique plus prosaïque mais plus facile à mettre en œuvre : le *mode souillé (taint mode)*, qui déclenche des mesures de sécurité particulières. Le mode souillé est activé automatiquement dans

certaines situations, par exemple lorsque les droits d'accès du programme et de l'utilisateur effectif sont discordants, il peut aussi être activé explicitement, ce qui est fortement conseillé pour les programmes serveurs.

En mode souillé, Perl active des *contrôles de pollution (taint checks)* : certains sont classiques, tels que l'interdiction d'écrire dans les répertoires du chemin de recherche des programmes exécutables.

En mode souillé, Perl affecte d'une marque spéciale toutes les données qu'un programme reçoit de l'extérieur, tels les champs de formulaires remplis directement par un internaute et qui pourraient comporter des injections de code (cf. la section 3 page 61) ; de même sont marqués souillés les arguments de ligne de commande, les variables d'environnement, et toutes les données lues depuis un fichier. Les variables souillées sont soumises à des restrictions d'usage : elles ne peuvent pas être utilisées pour modifier une donnée extérieure au programme, sauf si elles ont été dûment vérifiées et explicitement « blanchies ». Les meilleures méthodes de blanchissage de données reposent sur la technique du *filtrage* ; il s'agit ici du filtrage dans l'acception qui désigne une technique de programmation nommée en anglais *pattern matching*, à ne pas confondre avec le filtrage sur les réseaux, bien que celui-ci puisse recourir à celle-là. Bref, le filtrage évoqué ici consiste à tenter de détecter, dans le contenu de la variable souillée, des caractères ou « mots » potentiellement dangereux.

Séparation des privilèges dans le système

Nous avons déjà évoqué la problématique de la séparation des privilèges à la section 2 page 40, notamment du point de vue de la gestion des comptes des utilisateurs. Nous allons préciser ici quelques aspects un peu plus techniques.

Il est important que chaque utilisateur, à chaque instant, possède les privilèges qui lui sont indispensables pour accomplir son travail, et seulement ceux-là. S'il doit pour une opération particulière élever son niveau de privilèges, cette élévation doit être temporaire, et son effet doit être limité à l'opération en question. Cela est bien sûr encore plus impératif lorsqu'il s'agit des privilèges du super-utilisateur, `root` sous Unix ou `Administrateur` sous Windows, par exemple. Mais ces précautions déjà signalées ne suffisent pas.

La plupart des systèmes d'exploitation modernes administrent également la séparation des privilèges au sein même de l'espace mémoire affecté à chaque utilisateur ou, pour être plus précis, à chaque processus en cours d'exécution sur le système. Cet espace de mémoire (aussi dénommé espace adresse) est divisé en régions, chacune caractérisée par son contenu.

Par exemple, sous un système Unix tel que Linux, la zone de mémoire allouée à un programme en cours d'exécution est divisée en sections : le texte du code exécutable proprement dit est dans la section `.text`, les données initialisées sont dans la section `.data`, les données non initialisées dans la section `.bss`, sans préjudice des zones allouées au tas *(heap)* et à la pile *(stack)*. Chacune de ces sections, ainsi que le tas et la pile, sans oublier les zones affectées respectivement au code et aux données de chaque bibliothèque utilisée par le programme, seront installés dans des régions de mémoire particulières et bien identifiées, dotées de privilèges adéquats. Ainsi, seul le texte des régions affectées au code exécutable sera habilité à être exécuté, c'est-à-dire qu'une instruction de branchement ne sera valide que si son adresse de destination est contenue dans une telle région, sinon elle déclenchera une erreur et l'interruption du programme. De cette façon, il sera impossible d'exécuter du code placé dans une section de données ou sur la pile. De même, ne pourront être modifiées par programme que les données contenues dans les régions destinées à cet effet. De telles mesures de protection de la mémoire sont de nature à contrecarrer toute une famille de logiciels malfaisants, conçus à l'origine pour exploiter des débordements de tampon dans la pile (cf. section 5).

En effet, sur un système qui n'est pas doté de ce type de protection, il est possible d'injecter du code malfaisant dans une zone de mémoire disponible en écriture, puis de l'exécuter.

Architectures tripartites

La locution *architecture tripartite* traduit ici l'anglais *three tiers*, qui n'a jamais signifié trois-tiers (ni quatre-quarts), et qui pourrait aussi se traduire par architecture à trois niveaux (c'est le choix de Wikipédia) ou à trois étages. Il s'agit d'un modèle de construction de systèmes informatiques propre à en améliorer la sécurité, et nous n'hésiterons donc pas à recommander son usage. Il peut bien sûr y avoir plus de trois étages, et l'on parlera alors d'architecture multipartite, mais trois est

la cardinalité la plus fréquente. Il s'agit d'une extension du modèle client-serveur, fameux en son temps et aujourd'hui supplanté par les architectures construites à partir de serveur et de navigateurs Web.

L'idée de tripartition d'une application informatique consiste à séparer trois groupes de fonctions et à les implanter sur des systèmes informatiques différents, placés sur des réseaux distincts régis par des règles de sécurité spécifiques et adaptées à chacune de ces fonctions, qui sont :

1. l'interface utilisateur, nommée ici le *niveau présentation*, qui, de plus en plus souvent, sera affichée par un navigateur Web ;
2. les logiciels de traitement, qui constituent le *niveau logique*, éventuellement invoqués par l'intermédiaire d'un serveur Web ou d'un mandataire applicatif ;
3. le stockage des bases de données, qui constitue le *niveau données*.

Outre les avantages habituels de la modularité des applications, qui permettent notamment de faire évoluer les logiciels des postes de travail sans impact sur le système de bases de données, cette architecture améliore la sécurité globale du système : les bases de données, qui sont généralement la partie la plus sensible de l'ensemble, seront placées dans un sous-réseau hautement protégé, isolé de tout accès direct des utilisateurs ; les logiciels de traitement seront implantés sur des ordinateurs dépourvus de données, ce qui réduira les conséquences de leur éventuelle compromission.

Entre le navigateur de l'utilisateur et la couche traitement, on interposera souvent un mandataire applicatif, nommé en anglais *reverse proxy*, c'est-à-dire un serveur Web spécialisé qui analyse les requêtes émises par les utilisateurs, rejette les requêtes non autorisées ou malformées et réécrit les requêtes convenables pour les transmettre au logiciel d'application. Un tel système réduit de façon drastique le risque d'attaque réussie contre le serveur, à condition, certes, que les logiciels aient été écrits en respectant la règle du privilège minimum nécessaire, que les barrières soient bien baissées là où il le faut... et au prix d'un budget supplémentaire non négligeable. L'architecture tripartite devient ainsi quadripartite (cf. page 123).

6

Sécurité du réseau

La sécurité de l'informatique ne se limite certes pas à celle du réseau, mais il est indéniable que la plupart des incidents de sécurité surviennent par le réseau, et visent le réseau. Aussi le présent chapitre, qui lui est consacré, est-il le plus copieux du livre. Après un rappel des principes de l'Internet, nous traiterons des réseaux privés virtuels (VPN), du partage de fichiers à distance, des pare-feu, du Système de noms de domaines (DNS), des réseaux locaux virtuels (VLAN) et des réseaux sans fil (Wi-Fi).

Les protocoles poste à poste *(peer to peer)* et de téléphonie sur Internet seront abordés au chapitre 10, la détection d'intrusion au chapitre 11, les annuaires au chapitre 7.

Modèle en couches pour les réseaux

Avant de parler de sécurité des réseaux, il peut être utile de rappeler brièvement le modèle qui sert à les décrire. Le lecteur familier de ces notions peut sans risque passer à la section suivante.

L'architecture en couches a été formulée par le chercheur néerlandais Edsger Wybe Dijkstra (1930–2002) dans un article fameux publié en mai 1968 par les CACM, « *The structure of the THE multiprogramming system* » [41], pour représenter des systèmes qui relèvent simultanément de plusieurs niveaux d'abstraction. L'idée est d'isoler chaque niveau d'abstraction pertinent pour le système considéré, de façon à s'en faire une idée plus simple. Le principe de l'architecture en couches peut être rapproché de celui d'*architecture tripartite* que nous avons étudié à la page 102 ; le but en est le même : diviser un problème en sous-problèmes plus simples, isoler différents niveaux d'abstraction.

Application du modèle à un système de communication

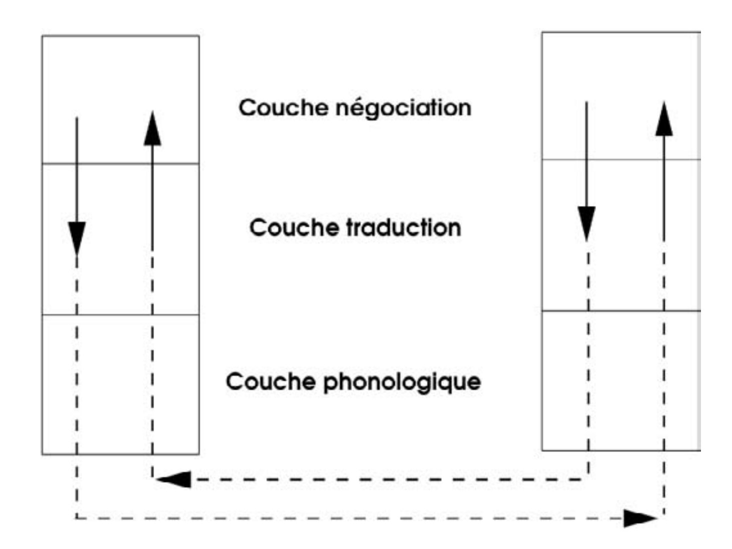
Ainsi, imaginons le système de communication constitué par deux chefs d'État, accompagnés de leurs interprètes respectifs, en train de mener une négociation bilatérale. Chaque chef d'État s'exprime dans sa langue nationale, que nous appellerons respectivement langue A et langue B ; chaque interprète traduit ce que dit son chef d'État en picto-saintongeais, langue diplomatique internationale. Nous pouvons modéliser ce système de communication au moyen de trois couches, illustrées par la figure 6.1 :

- **la couche 3** « négociation » décrit les interactions entre les deux chefs d'État ;
- **la couche 2** « traduction » décrit les interactions entre les interprètes ;
- **la couche 1** « phonologique » décrit les mécanismes physiologiques et physiques en jeu dans la communication verbale entre deux êtres humains.

Chacune de ces couches a sa logique propre, qu'il est possible d'étudier sans se préoccuper des deux autres, bien que la couche 3 ne puisse pas exister sans les couches 2 et 1, ni la couche 2 sans la couche 1.

Lors de leur négociation, les deux chefs d'État respectent un *protocole*, que nous nommerons protocole de la couche 3 : il comporte les règles du savoir-vivre des grands de ce monde, ainsi peut-être que les traités conclus entre les pays dont ils sont les dirigeants.

Figure 6.1
Un système de communication à trois couches



De leur côté, les interprètes respectent un protocole de couche 2, qui comporte sûrement aussi des règles de savoir-vivre (sans doute un peu moins cérémonieuses que celles de la couche 3), mais surtout le respect des règles de la grammaire du picto-saintongeais et de la sémantique de son lexique, nécessaires à la bonne compréhension entre les négociateurs.

La couche 1 pourrait être décrite en termes de larynx, de cordes vocales, de propagation d'ondes sonores dans l'atmosphère.

Notons qu'après avoir prononcé une phrase, disons en langue A, le chef d'État considéré doit attendre que son interprète l'ait traduite en picto-saintongeais, puis que l'interprète de l'autre partie l'ait traduite de picto-saintongeais dans la langue B de l'autre chef d'État, et vice-versa, sans oublier les délais de propagation des sons qui constituent les paroles dans l'atmosphère (dans le vide, ou sur la lune, ce dispositif échouerait, il faudrait en imaginer un autre).

Alors que les échanges entre deux interlocuteurs d'une couche donnée sont régis par un protocole, les échanges entre un chef d'État et son interprète sont régis par une *interface*, ainsi, de façon générale, que les échanges entre un agent de la couche n et un agent de la couche $n-1$, du même côté de la communication. Pour que la

négociation se déroule de façon satisfaisante, il ne faut pas qu'il y ait d'interactions directes entre un agent de la couche n d'une partie et un agent de la couche $n-1$ de l'autre partie : si le chef d'État de langue nationale A s'adresse directement à l'interprète du chef d'État de langue nationale B dans une langue C, différente du picto-saintongeais, langue internationale, on risque l'incident diplomatique.

Mais, quoi qu'il en soit des couches 1 et 2, les diplomates qui rédigeront le communiqué final et les journalistes qui commenteront l'entrevue ne prendront en considération que les échanges de la couche 3, ils *feront abstraction* des échanges des couches basses, pourtant indispensables. Et tel était bien le but poursuivi.

Modèle ISO des réseaux informatiques

Les réseaux informatiques ont fait l'objet d'une modélisation par l'ISO selon un modèle en sept couches nommé OSI (pour *Open Systems Interconnection*), qui n'a pas eu beaucoup de succès en termes de réalisations effectives, mais qui s'est imposé par sa clarté intellectuelle comme le meilleur outil de conceptualisation des réseaux. La figure 6.2 représente les quatre couches basses du modèle ISO ; les couches 5 à 7 sont moins intéressantes et peu évoquées par la littérature ; mentionnons néanmoins la couche 7, « Application », qui correspond aux logiciels utilisés directement par les utilisateurs, tels que navigateur Web, logiciel de courrier électronique ou de connexion à distance.

La couche 1 concerne la mise en œuvre du support physique de la communication, il s'agit d'électronique et de traitement du signal.

L'objet de la couche 2 (dite *liaison de données*) est d'acheminer de façon sûre des données entre deux stations directement connectées au même support physique. L'ensemble de données élémentaire véhiculé par la couche 2 s'appelle une *trame*.

La couche 3 (dite *réseau*) envisage deux stations connectées à des réseaux différents, eux-mêmes reliés à d'autres réseaux qui forment un *Internet*. Il faut, à travers un réseau de réseaux interconnectés, un peu comme les réseaux ferroviaires européens entre eux, trouver un itinéraire pour acheminer les données : c'est la question du *routage*. L'ensemble de données élémentaire véhiculé par la couche 3 s'appelle un *paquet*.

La couche 4 *(transport)* vise à assurer entre deux stations distantes l'acheminement sûr des données de bout en bout par un itinéraire calculé par la couche 3, soit à

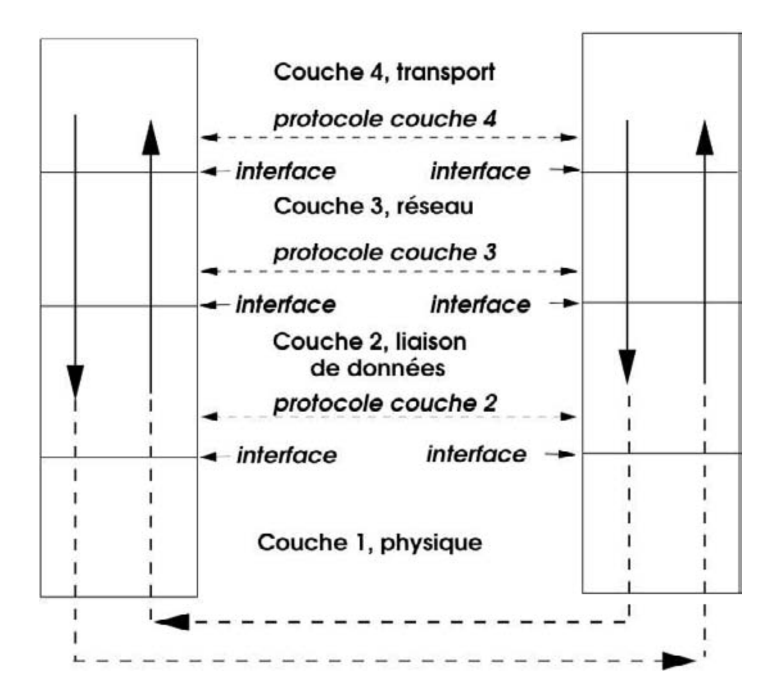


Figure 6.2
Les quatre couches basses du modèle ISO

établir entre ces deux stations distantes le même type de communication qui serait assuré par la couche 2 si elles partageaient le même support physique, comme si le réseau complexe qui les sépare était un support unique. L'ensemble de données élémentaire véhiculé par la couche 4 s'appelle un *segment*.

Cette abstraction en couches permet de concevoir un réseau comme l'Internet, fondé sur les couches 3 et 4, indépendant des couches basses (1 et 2), et disponible pour toutes sortes d'applications non prévues. Il en a été ainsi parce que les créateurs de l'Internet, qui utilisaient pour construire le réseau une infrastructure téléphonique dont les opérateurs ne leur révélaient pas les caractéristiques internes, ont fait de nécessité vertu : l'architecture qu'ils ont imaginée peut fonctionner « au-dessus » de n'importe quelle infrastructure de communication. C'est ce qui a assuré le succès de l'Internet, et lui a permis, au fil des années, de s'adapter

aux nouveaux supports : fibre optique, liaisons satellite, liaisons sans fil, infrarouge, sans que ses protocoles en soient affectés.

> **POUR ALLER PLUS LOIN**
>
> On consultera avec profit à ce sujet la contribution de Jean-François Abramatic. « Croissance et évolution de l'Internet ». Dans *Université de tous les savoirs – Les Technologies*, volume 7, Paris, 2002. Odile Jacob. Cf. aussi le livre classique de Katie Hafner et Matthew Lyon. *Where Wizards Stay Up Late – The Origins of the Internet*. Pocket Books, Londres, 1996.

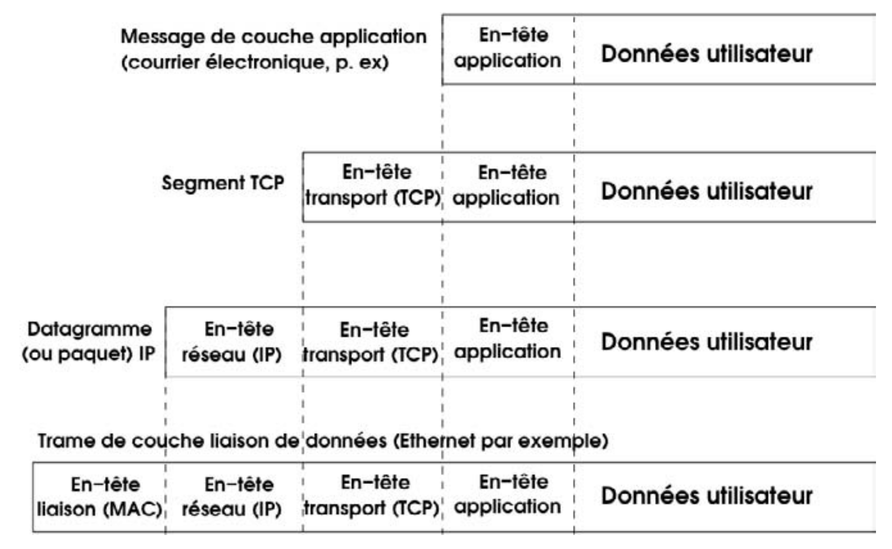
Une réalisation : TCP/IP

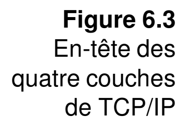
L'Internet est construit autour de son protocole de réseau, IP *(Internet Protocol)*, et de son principal protocole de transport, TCP *(Transmission Control Protocol)*. Le protocole IP correspond à la couche 3 du modèle OSI, la couche réseau. La « pile » TCP/IP (comme une pile de couches... empilées) n'obéit pas strictement à la nomenclature du modèle OSI : elle comporte une couche liaison de données qui englobe les couches 1 et 2 de l'OSI, la couche IP (réseau) correspond à la couche 3 de l'OSI, la couche TCP[1] (transport) correspond à la couche 4 de l'OSI. La couche « applications » englobe tout ce qui relève des couches hautes de l'OSI.

L'architecture de TCP/IP peut être vue sous l'angle suivant. À partir d'un message émis par un utilisateur, chaque couche en partant de la plus haute lui ajoute des en-tête qui contiennent les informations nécessaires à son fonctionnement, ce que montre la figure 6.3 page suivante.

Ainsi, un message électronique sera d'abord doté par votre logiciel de courrier des en-tête applicatifs, en l'occurrence tels que décrits par la RFC 822 révisé en 2822 (ce sont les lignes `From :`, `To :`, `Subject :`, etc. qui figurent en tête des messages). Votre logiciel de courrier mettra également, si besoin est, le contenu de message en forme, toujours afin d'adhérer à ce standard qu'est le RFC 2822. Un message complexe, par exemple mélangeant texte et image, pourra être formaté selon le standard MIME.

[1]... ou UDP, autre couche transport disponible au-dessus d'IP.



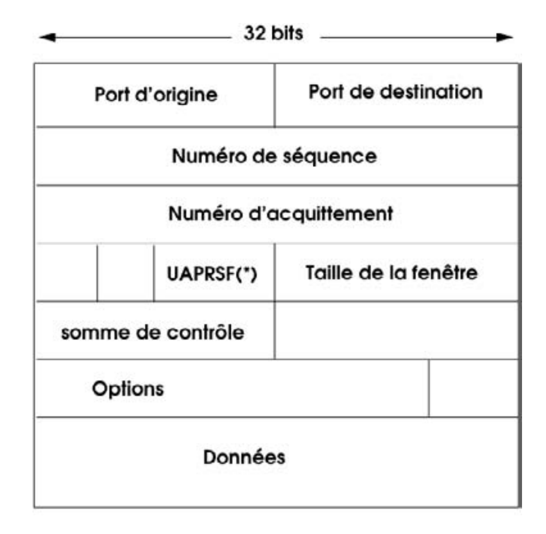

Figure 6.3
En-tête des quatre couches de TCP/IP

> **VOCABULAIRE Les RFC**
>
> Les *Requests for Comments (RFC)* sont les documents de référence pour le fonctionnement du réseau. Citons ici le nom de Jon Postel, éditeur des RFC depuis la première, en 1969, jusqu'à sa mort en 1998, et auteur ou coauteur de 204 d'entre elles, ce qui lui a conféré une influence considérable sur la physionomie du réseau. Toutes les RFC sont accessibles par l'URL *(Universal Resource Locator)* `http://www.ietf.org/rfc/` ou sur de nombreux sites miroirs. Nous ne saurions trop en conseiller la lecture ; même si la qualité de leur style est inégale, elles fournissent sur l'Internet une information de première main, souvent exposée très clairement, et dont la citation dans les dîners en ville vous assurera une réputation de gourou du réseau.

Après ces transformation, ce message, devenu conforme à la RFC 2822, doit être transporté afin de parvenir jusqu'à son destinataire légitime. Pour cela, votre système de courrier électronique s'appuie un autre standard, le protocole SMTP défini dans la RFC 2821 (qui met à jour la RFC 821). Le protocole SMTP décrit les échanges qui vont se dérouler entre les parties (le système de messagerie de l'expéditeur et celui du destinataire).

C'est cet ensemble structuré qui représente les « données utilisateur » du protocole SMTP et qui sera découpé en segments TCP, chacun doté de l'en-tête convenable décrit à la figure 6.4.

Figure 6.4
En-tête de segment TCP



Chaque segment TCP sera empaqueté dans un ou plusieurs paquets IP, qui possèdent chacun un en-tête, décrit à la figure 6.5 page suivante pour la version 4 du protocole IP, et à la figure 6.6 page ci-contre pour la version 6. Et chaque paquet sera expédié sur la couche liaison de données qui correspond au support physique, Ethernet par exemple.

Le protocole réseau IP fournit à la couche transport un service non fiable non connecté de datagrammes. Le terme datagramme signifie que le flux de bits remis par la couche transport (TCP) est découpé en morceaux (les datagrammes) acheminés indépendamment les uns des autres. En général les datagrammes sont transmis en entier sur le réseau, mais le protocole prévoit que tous les segments du réseau n'admettent pas forcément la même taille de données à transmettre, auquel cas il peut arriver qu'un datagramme soit découpé en plusieurs *paquets*, mais la plupart du temps un datagramme correspond à un paquet, les deux termes sont

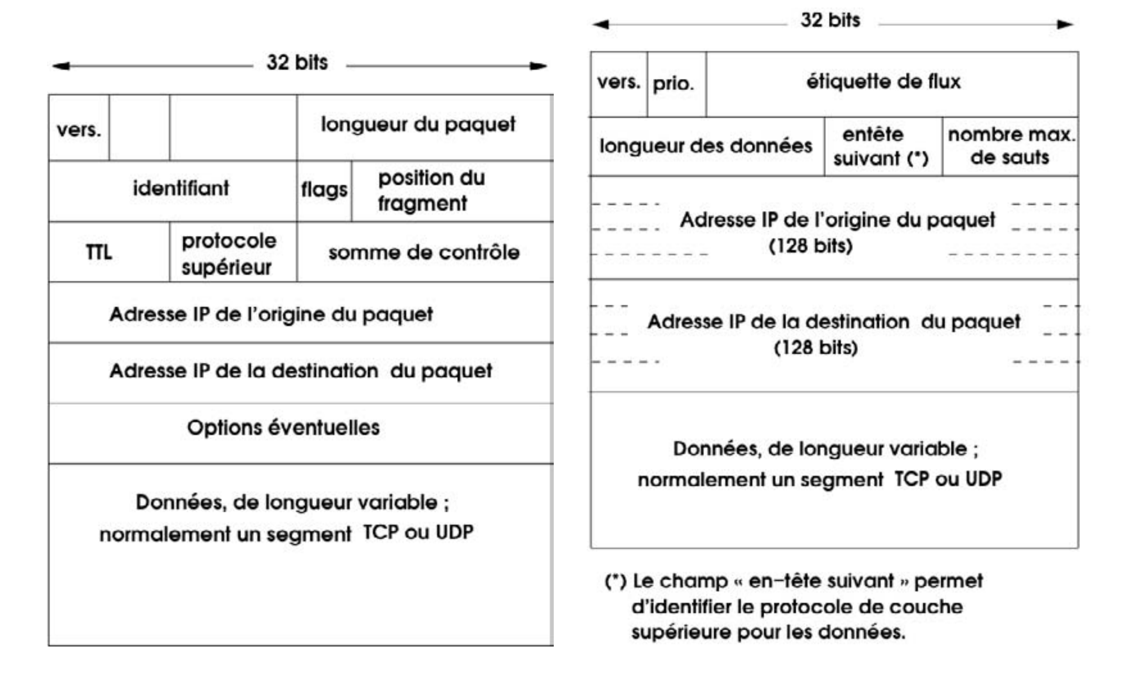


Figure 6.5
En-tête de paquet IPv4

Figure 6.6
En-tête de paquet IPv6

quasiment synonymes. Notons cependant que cette possibilité de fragmenter les datagrammes est parfois utilisée par les pirates pour dissimuler leurs attaques.

Par « non fiable » nous entendons que la couche IP ne fournit aucune garantie de remise des datagrammes ni aucun contrôle d'erreur, et par « non connecté » nous entendons que la couche IP ne maintient aucune information d'état sur une transmission de données en cours, et notamment qu'elle ne garantit pas la remise des datagrammes dans l'ordre dans lequel ils ont été émis.

Ces caractéristiques sont de nature à inquiéter les néophytes, et semblent curieuses, d'autant plus que la couche de liaison de données fournit à la couche réseau, pour chaque segment physique d'un chemin de données utilisé par un datagramme, un service fiable de flux de bits remis dans le bon ordre.

En fait, la couche IP ne fournit pas de contrôle d'erreur parce que de toute façon la couche TCP devra en effectuer, ainsi que la vérification du bon ordre de remise des datagrammes au terminus de la transmission, et que de tels contrôles au niveau de la couche 3 seraient redondants. Son ascétisme et sa désinvolture confèrent à la couche IP la simplicité, la légèreté et la souplesse qui font son efficacité.

Les réseaux privés virtuels (VPN)

Au chapitre 4 page 71 nous avons décrit l'usage de techniques cryptographiques pour le chiffrement de messages individuels, mais ce n'est en aucun cas le seul usage de cette technique. On peut imaginer, et de plus en plus c'est ce qui sera réalisé, le chiffrement systématique de toutes les communications en réseau.

Si l'on procède ainsi, chiffrer message par message serait très inefficace : on choisira plutôt de chiffrer le flux de l'ensemble du trafic sur un ou plusieurs itinéraires donnés, cela constituera un *réseau privé virtuel*, ou VPN, comme *Virtual Private Network*. Il s'agira par exemple d'établir un canal chiffré entre deux nœuds quelconques de l'Internet, ces nœuds pouvant eux-mêmes être des routeurs d'entrée de réseaux. On aura ainsi établi une sorte de tunnel qui, à travers l'Internet, reliera deux parties éloignées l'une de l'autre du réseau d'une même entreprise pour donner l'illusion de leur contiguïté. Mais le chiffrement permet aussi d'établir un VPN personnel pour un utilisateur, par exemple entre son ordinateur portable et le réseau local de l'entreprise.

Principes du réseau privé virtuel

Le chiffrement est généralement utilisé pour les VPN de la façon suivante : l'algorithme de Diffie-Helmann est utilisé pour procéder au choix d'un secret partagé, qui constituera une clé de session pour chiffrer le trafic, et qui sera renouvelé à intervalles réguliers.

Il y a en revanche une assez grande variété de solutions pour introduire le VPN dans l'architecture du réseau :

- **Couche 3 :** introduire le VPN au niveau de la couche réseau (n° 3 du modèle ISO) semble la solution la plus logique : il s'agit bien de créer un *réseau virtuel*, après tout. C'est la solution retenue par la pile de protocoles désignés

collectivement par l'acronyme IPSec, que nous décrirons à la section suivante. Les protocoles IPSec sont implantés dans le noyau du système d'exploitation, ce qui assure une plus grande sûreté de fonctionnement (face aux attaques notamment) et de meilleures performances (un protocole implanté en espace utilisateur passe son temps à recopier des tampons de mémoire entre l'espace noyau et l'espace utilisateur).

- **Couche 4 :** La disponibilité de bibliothèques SSL/TLS (pour *Secure Socket Layer/Transport Layer Security*) à la mise en œuvre facile a encouragé le développement de VPN de couche 4 (transport), comme *OpenVPN* ou les tunnels SSL[2]. *OpenVPN*, par exemple, établit un tunnel entre deux stations, et par ce tunnel de transport il établit un lien réseau, chaque extrémité recevant une adresse IP.
- **Couche 7 :** Le logiciel SSH *(Secure Shell)*, qui comme son nom l'indique est un client de connexion à distance chiffrée, donc de couche 7, permet de créer un tunnel réseau.
- **Couche 2 :** Mentionnons ici pour mémoire les réseaux locaux virtuels (VLAN), que nous étudierons plus en détail à la page 156 : il ne s'agit pas à proprement parler de VPN, mais souvent ils ont un même usage : regrouper les stations d'un groupe de personnes qui travaillent dans la même équipe sur un réseau qui leur soit réservé, séparé des réseaux des autres équipes.

  L2TP *(Layer Two Tunneling Protocol)*, comme son nom l'indique, encapsule une liaison de couche 2 (liaison de données) sur un lien réseau (couche 3).

IPSec

IPSec désigne un ensemble de RFC destinées à incorporer les techniques de chiffrement (et d'autres, relatives aussi à la sécurité) au protocole IP lui-même, plutôt que d'avoir recours à des solutions externes. IPv6 a été conçu pour comporter d'emblée toutes les spécifications IPSec.

IPSec comporte essentiellement deux protocoles :

- le protocole AH *(Authentification Header)* assure l'authenticité et l'intégrité des données acheminées ; c'est un protocole réseau, de couche 3 donc, que l'on peut voir comme une option d'IP ;

[2] Cf. `http://openvpn.net/`

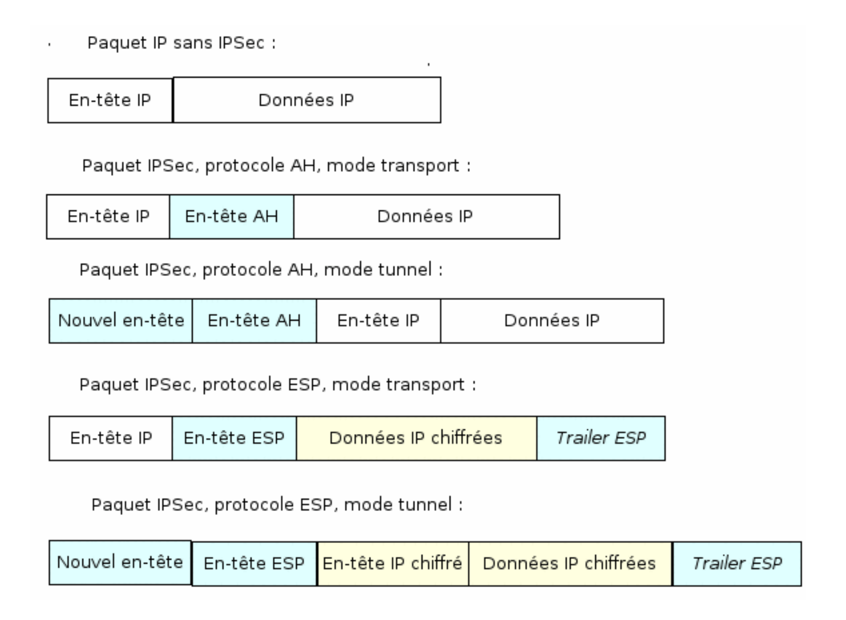
- le protocole de transport ESP (couche 4) *(Encapsulating Security Payload)* assure la confidentialité et l'intégrité des données, leur authenticité étant assurée de façon optionnelle.

Avec l'un ou l'autre de ces protocoles, IPSec peut fonctionner en *mode transport* ou en *mode tunnel* :

- en mode tunnel chaque paquet IP est encapsulé dans un paquet IPSec lui-même précédé d'un nouvel en-tête IP ;
- en mode transport un en-tête IPSec est intercalé entre l'en-tête IP d'origine et les données du paquet IP.

La figure 6.7 illustre ces différentes possibilités.

Figure 6.7 Protocoles et modes IPSec



Les protocoles AH et ESP sont complétés par le protocole d'échange de clés IKE *(Internet Key Exchange)*, défini dans la RFC 2409, et par le protocole de gestion de clés ISAKMP *(Internet Security Association and Key Management Protocol)*, défini dans la RFC 2408. Il semble bien qu'ISAKMP soit un protocole irrémédiable-

ment mal conçu, qu'il n'y a plus qu'à réécrire — le travail est en cours mais risque d'être comparable au tissage du voile de Pénélope.

On l'aura compris, IPSec est une usine à gaz, son implémentation complète est de ce fait difficile. Pour voir son déploiement à grande échelle il faudra sans doute attendre la généralisation d'IPv6. Cela étant dit, pour le déploiement de VPN sur une infrastructure (par opposition aux VPN à usage personnel), c'est la solution qu'il faut retenir.

Autres réseaux privés virtuels

En attendant la stabilisation d'IPSec, il existe d'autres procédés pour créer des réseaux privés virtuels, intégrés à TCP/IP de façon peut-être moins satisfaisante, mais plus pratique.

1. L2TP (*Layer Two Tunneling Protocol*, RFC 2661), comme son nom l'indique, encapsule une liaison de couche 2 (liaison de données) sur un lien réseau (couche 3), ce qui permet à un PC distant d'avoir accès au réseau de son entreprise comme s'il était connecté au réseau local, et ainsi d'avoir accès aux serveurs de fichiers, aux imprimantes, etc.
2. MPLS (*Multi-Protocol Label Switching*, RFC 2547) est un protocole de niveau 3 (réseau) qui permet d'établir un tunnel privé au sein d'un réseau public ; il est surtout utilisé par les fournisseurs d'accès à l'Internet pour proposer à leurs clients un moyen de créer un réseau privé entre plusieurs sites d'une même entreprise.
3. Mentionnons également des procédés pour créer des tunnels dits « IP dans IP » (couche 3, réseau) par divers procédés, ou encore les réseaux virtuels créés au moyen de TLS *(Transport Layer Security)*, qui, comme son nom l'indique, est une version de la couche 4 renforcée du point de vue de la sécurité.

Aujourd'hui, la plupart des VPN effectivement en fonction appartiennent à la dernière catégorie de la liste ci-dessus. Il existe des boîtiers qui contiennent des systèmes tout configurés, qu'il suffit de placer derrière les routeurs d'entrée de réseau pour disposer d'un VPN entre deux sites.

Il convient, avant de clore cette section, de signaler que si la technique des réseaux locaux virtuels *(Virtual Local Area Network, VLAN)* vise un objectif en principe assez différent de celui des VPN, elle peut dans certains cas être envisagée

comme une solution de substitution. Nous évoquerons cette technique plus loin, à la page 156.

Comparer les procédés de sécurité

Avant d'utiliser un procédé de sécurité, il faut se faire une idée assez précise de ce que protègent, et de ce que ne protègent pas, les différents usages possibles du chiffrement :

- Si Aldebert envoie à Barkissa un message électronique chiffré avec la clé publique *GnuPG* de Barkissa, seule Barkissa pourra lire le message ; même si Aldebert par erreur a également envoyé le message à Charlotte, celle-ci ne pourra pas le déchiffrer. Si un jour un juge d'instruction veut lire ce message qui en fait contenait le plan d'un complot pour renverser l'État, il devra obtenir la clé privée de Barkissa pour lire le contenu de sa boîte à lettres.
- Si Aldebert et Barkissa ne chiffrent pas les messages électroniques qu'ils échangent, mais que les serveurs de messagerie des réseaux locaux de leurs employeurs respectifs utilisent les versions sécurisées par TLS des protocoles de courrier électronique SMTP et POP, les échanges seront chiffrés pendant la circulation des messages sur le réseau, mais pas sur leurs postes de travail. Les messages seront ainsi protégés contre l'espionne Charlotte si celle-ci a accès au réseau, mais pas dans leurs boîtes à lettres respectives si Charlotte a accès à leurs ordinateurs. De même, le juge d'instruction, lorsqu'il aura saisi les disques durs, pourra lire les messages sans difficultés.
- Aldebert et Barkissa peuvent aussi disposer de VPN IPSec pour accéder aux réseaux qui abritent leurs serveurs de messagerie respectifs, qui ne sont accessibles que par ce procédé : ce dispositif garantit que l'espionne Charlotte ne pourra pas accéder à ces serveurs, car seuls les utilisateurs autorisés et authentifiés par IPSec le peuvent. Ce type d'accès permet à Aldebert, à Barkissa et à leurs collègues d'accéder ainsi à tous les services du réseau local, pas uniquement à la messagerie.
- Les serveurs de messagerie respectifs d'Aldebert et de Barkissa pourraient aussi être accessibles par un dispositif de type *Webmail*, sécurisé par

HTTPS, ce qui est une variante de la solution 2, où le message lu reste sur le serveur.

VOCABULAIRE SMTP et POP

Simple Mail Transport Protocol est le protocole d'échange de messages électroniques entre serveurs de messagerie, il est également utilisé par les logiciels de courrier électronique sur les postes de travail pour l'envoi des messages. La version sécurisée s'appelle ESMTP.

Post Office Protocol est le protocole utilisé sur un poste de travail pour relever une boîte à lettres sur un serveur de messagerie distant et transférer les messages sur le poste de travail. La version sécurisée s'appelle POP3S.

Partager des fichiers à distance

Si l'on désire utiliser à distance (à travers l'Internet) certains protocoles intrinsèquement non sûrs, l'établissement d'un réseau privé virtuel est le seul moyen acceptable du point de vue de la sécurité.

Le type même du protocole non sûr est le protocole de partage de fichiers : comme son nom l'indique, un tel protocole permet à plusieurs utilisateurs, depuis leurs postes de travail, d'accéder à des fichiers emmagasinés sur un serveur distant, exactement comme s'ils étaient sur leur disque dur local (enfin, presque exactement). De tels protocoles ont nom :

- *AppleShare* sur Macintosh ;
- *Network File System* (NFS) dans l'univers Unix/Linux ;
- *Common Internet File System* (CIFS), *Server Message Block* (SMB) ou *Netbios* dans le monde *Microsoft Windows* ;
- et d'autres...

Les protocoles de partage de fichiers sont à proscrire dans un environnement non sûr pour les raisons suivantes :

- par définition, ils permettent d'exécuter sur une machine distante un programme capable de créer, de modifier, de lire ou de détruire un fichier ; il s'agit là d'actions puissantes et complexes ;
- pour des raisons où se mêlent la paresse des concepteurs et la recherche de meilleures performances, qui en est souvent le prétexte, il s'agit de *protocoles*

sans état, c'est-à-dire que chaque message qui contribue à une action, et une action sur un fichier à distance comporte l'échange de nombreux messages entre les deux ordinateurs, chaque message donc est indépendant du précédent et du suivant ;

- il découle de la caractéristique précédente qu'il est relativement facile pour un attaquant de s'immiscer dans une action légitime en cours pour y insérer ses propres messages, qui vont avoir pour but d'accomplir des actions non désirées par le propriétaire du ou des fichiers ;
- de fait, les protocoles de partage de fichiers ont fait l'objet d'innombrables publications de vulnérabilités dont beaucoup ne sont ni corrigées ni corrigibles ;
- lorsque l'on analyse les recherches que font les pirates sur le réseau (les *scans de port*) avec l'objectif de trouver des sites dont la fragilité pourrait être favorable à leurs projets, ce qu'ils recherchent avec le plus d'avidité, ce sont les machines qui offrent une porte ouverte sur un protocole de partage de fichiers, car c'est ce qu'il y a de plus facile à attaquer et à vaincre ;
- utiliser un protocole de partage de fichiers à travers l'Internet sans autre protection qu'un identifiant et un mot de passe pour accéder au serveur, protection largement illusoire ici, est proprement suicidaire.

Outre ces raisons techniques, il y a des raisons tout simplement pragmatiques (nous pourrions dire aussi fonctionnelles, ou logiques) de ne pas utiliser de façon trop laxiste un protocole de partage de fichiers. La question des droits d'accès (lecture, écriture, création, destruction) doit faire l'objet de précautions particulières. Les règles de bon sens sont les suivantes : un serveur de partage de fichiers peut être ouvert soit en lecture et en écriture exclusivement pour chaque propriétaire de chaque fichier, soit pour plusieurs personnes, mais alors en lecture seule, sinon il en résulte des accidents tels que corruption de fichiers ou « perte » des droits d'accès. Si on réfléchit un peu en gardant à l'esprit la caractère « sans état » du protocole, c'est logique, et l'expérience confirme ici largement le raisonnement logique : aucun dispositif ne peut empêcher que deux utilisateurs qui auraient ouvert directement sur le serveur un même document de traitement de texte pour le modifier chacun de son côté ne le sauvegardent concurremment ; dans le meilleur des cas les modifications effectuées par un des auteurs seront perdues, et dans bien des

cas le document sera corrompu et irrécupérable ; les droits d'accès pourront aussi être altérés de façon peu prévisible par les manœuvres d'un auteur.

VOCABULAIRE Un *port*

Un *port* (en français *sabord*, orifice destiné à laisser passer un flux) dans la terminologie TCP/IP est un numéro conventionnel qui sera associé d'une part à un type de trafic (caractérisé par le protocle utilisé), d'autre part à une adresse IP. Le couple adresse IP – numéro de port définit de que l'on appelle une *socket*, que l'on pourrait traduire par *prise*. Une *socket* identifie *de façon unique* une extrémité de connexion.

Par convention certains numéros de ports sont réservés aux serveurs de certains protocoles ; ainsi le port 80 est réservé au protocole HTTP (Web), le port 25 à SMTP (courrier électronique), les ports n° 137, 138 et 139 au protocole de partage de fichiers *Netbios*, c'est-à-dire qu'un *serveur* Netbios sera en écoute sur le réseau et attendra des tentatives de connexion sur ces numéros de port, cependant que les *clients* Netbios essaieront de s'y connecter.

À l'extrémité côté client, le numéro de port est quelconque, en général supérieur à 1024, et unique pour son adresse IP. La connexion est ainsi identifiée de façon unique par le quadruplet {adresse IP d'origine, port d'origine, adresse IP de destination, port de destination}. Cette abstraction permet à un nœud unique du réseau d'être simultanément serveur pour plusieurs protocoles, et également d'être à la fois serveur et client. Les pirates recherchent activement sur l'Internet les machines accessibles qui laissent ouverts des ports de protocoles réputés vulnérables pour essayer de compromettre le serveur à l'écoute.

Sécuriser un site en réseau

Assurer la sécurité de systèmes informatiques abrités sur un site connecté à l'Internet et de ce fait accessible du monde entier est une autre branche de ce domaine dont, en 2006, beaucoup d'organisations ne semblent pas avoir pris l'exacte mesure.

Comme nous l'avons mentionné, il appartient tout d'abord aux responsables du site de déterminer le périmètre qu'ils veulent protéger, ainsi que ce qu'ils veulent autoriser.

Cet examen aboutit généralement à identifier un certain nombre de services qui doivent être accessibles de l'extérieur par nature, comme un serveur Web, un relais de courrier électronique, un serveur DNS. Les ordinateurs qui abritent ces

services devront être visibles de l'Internet, c'est-à-dire que le DNS doit publier leurs adresses.

Les autres ordinateurs, que ce soient des serveurs internes ou les stations de travail des personnes qui travaillent sur le site, ne doivent pas être visibles, mais il faut néanmoins qu'ils puissent accéder à l'Internet. En d'autres termes, une session TCP initiée de l'intérieur du site depuis un de ces ordinateurs est autorisée, mais une session initiée de l'extérieur vers le même ordinateur est interdite parce que réputée erronée ou hostile (sous l'angle de la sécurité les deux termes sont quasiment synonymes).

De quels moyens disposons-nous en 2006 pour mettre en œuvre une telle politique de sécurité ? Il nous faut pour cela rappeler le fonctionnement des réseaux, et notamment ce qui concerne les données contenues dans les en-tête de datagrammes IP et de segments TCP, ainsi que le routage.

Segmentation

Chaque paquet IP qui se présente à un routeur est doté d'une fiche signalétique constituée de ses en-tête. Les informations principales, sous l'angle qui nous intéresse ici, sont les adresses IP d'origine et de destination et le protocole de transport (TCP ou UDP), figurant dans l'en-tête de datagramme IP, et les numéros de ports[3] d'origine et de destination, figurant dans l'en-tête de segment TCP ou de datagramme UDP. La mention dans l'en-tête IP du protocole de transport permet de connaître le format de l'en-tête de transport (TCP ou UDP), et ainsi d'y retrouver le numéro de port. Nous avons vu que l'association d'une adresse et d'un port constituait une *socket*. Une paire de *sockets* identifie de façon unique une connexion dans le cas de TCP. Le routeur maintient une table des connexions TCP établies qui lui permet de déterminer si ce paquet appartient à une communication déjà en cours (parce qu'établie entre les mêmes adresses IP et avec les mêmes numéros de ports, par exemple) ou à une nouvelle communication. Bien sûr, les adresses d'origine et de destination du paquet permettent au routeur de déterminer le sort qui lui sera réservé :

- délivrance directe au nœud destinataire sur un réseau local auquel le routeur est directement connecté ;

[3] Voir l'encadré page 121 pour la définition du *port*.

- émission vers un autre routeur, par une interface de sortie du réseau local, soit parce que les tables de routage disent que le réseau de destination est accessible par ce chemin, soit parce que c'est le routeur de sortie par défaut ;
- mise à la poubelle.

Le routage est un important instrument de sécurité. Il permet de découper un grand réseau en autant de sous-réseaux qu'on le souhaite, et de contrôler le trafic entre ces sous-réseaux. Les sous-réseaux peuvent d'ailleurs être virtuels, pour s'affranchir des contraintes de localisation, ce qui sera de plus en plus souvent le cas avec le développement de l'informatique mobile. Cela exige des compétences et du travail : ce que nous avons dit du routage montre que c'est tout sauf simple. Mais un tel investissement est indispensable à qui veut disposer d'un réseau sûr.

Filtrage

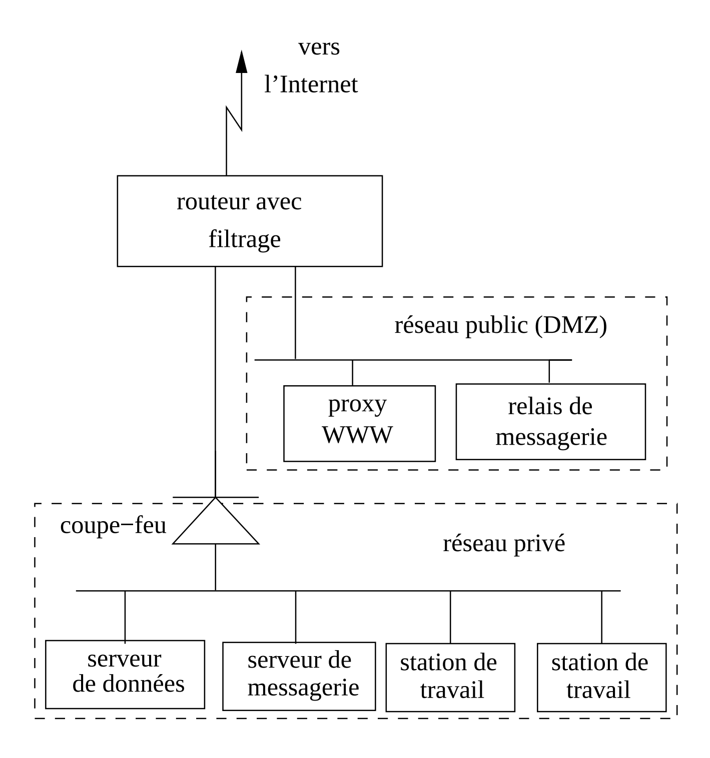
Forts de cette possibilité, nous pourrons segmenter le réseau de notre site en un sous-réseau public, qui abritera les serveurs visibles de l'extérieur, et un sous-réseau privé, éventuellement divisé lui-même en sous-réseaux consacrés à tel groupe ou à telle fonction. Chacun de ces sous-réseaux verra son accès et son trafic régis par des règles spéciales.

Les règles d'accès et de trafic appliquées aux réseaux consistent à établir quels sont les type de paquets (en termes de protocole et de numéro de port, en l'état actuel de la technique) autorisés en entrée ou en sortie depuis ou vers tel réseau ou telle adresse particulière. Ainsi un serveur de messagerie (appelé également passerelle de messagerie, ou MTA, comme *Mail Transfer Agent*) pourra recevoir et émettre du trafic SMTP (port 25) mais n'aura aucune raison de recevoir du trafic NNTP *(Network News Transfer Protocol)* sur le port 119. Appliquer ce genre de règles, c'est faire du *filtrage par port*.

Le sous-réseau public (souvent appelé « zone démilitarisée » ou DMZ) devra faire l'objet de mesures de sécurité particulièrement strictes, parce que de par sa fonction il sera exposé à toutes les attaques en provenance de l'Internet. Le principe de base est : tout ce qui n'est pas autorisé est interdit, c'est-à-dire que tout paquet qui n'a pas de justification liée aux fonctions du serveur de destination doit être rejeté.

Il est prudent que les serveurs en zone publique contiennent aussi peu de données que possible, et même idéalement qu'ils n'en contiennent pas du tout, pour éviter qu'elles soient la cible d'attaques. Ceci semble contradictoire avec le rôle même

Figure 6.8
Réseau avec DMZ et pare-feu



d'un accès à l'Internet, mais cette contradiction peut être résolue en divisant les fonctions. Ainsi pour un serveur de messagerie il est possible d'installer un relais en zone publique qui effectuera toutes les transactions avec le monde extérieur mais transmettra les messages proprement dits à un serveur en zone privée, inaccessible de l'extérieur, ce qui évitera que les messages soient stockés en zone publique en attendant que les destinataires en prennent connaissance. De même un serveur Web pourra servir de façade pour un serveur de bases de données en zone privée. Ces serveurs en zone publique qui ne servent que de relais sont souvent nommés serveurs mandataires, ou mandataires applicatifs, *proxy servers* en anglais (cf. page 102).

La figure 6.8 représente un tel dispositif, avec un routeur d'entrée qui donne accès à la DMZ, et un pare-feu (*firewall*), qui est en fait un routeur un peu particulier dont nous détaillerons le rôle ci-dessous, qui donne accès à un réseau privé.

Le relayage entre zone publique et zone privée fonctionne aussi dans l'autre sens : l'utilisateur en zone privée remet son courrier électronique au serveur privé, qui l'envoie au relais en zone publique, qui l'enverra au destinataire. Pour consulter une page sur le Web, l'utilisateur s'adresse au serveur relais qui émettra la vraie requête vers le monde extérieur. Ici le relayage peut procurer un autre avantage, celui de garder en mémoire cache les pages obtenues pendant un certain temps, pour les fournir au même utilisateur ou à un autre lorsqu'il les redemandera sans avoir à accéder à la source distante (une analyse statistique des requêtes révèle que dans un contexte donné les gens accèdent souvent aux mêmes pages).

Le filtrage par port permettra la communication entre le proxy et le serveur en zone privée de façon contrôlée. Les routeurs disposent de fonctions de filtrage assez élaborées, permettant de distinguer quels protocoles et quels numéros de ports sont autorisés selon l'origine et la destination, et si telle communication a été initiée depuis un nœud à l'intérieur ou à l'extérieur du réseau.

Pare-feu

La plupart des réseaux privés sont munis d'un pare-feu (*firewall*), ordinateur qui filtre les communications, un peu comme un routeur — d'ailleurs il est possible de configurer un routeur pour lui faire jouer le rôle d'un pare-feu simple. Un routeur doit décider au coup par coup du sort de chaque paquet, avec seulement une faible possibilité d'analyse historique, alors qu'un pare-feu efficace contre les attaques subtiles doit pouvoir faire des choses plus compliquées.

La configuration d'un pare-feu consiste à rédiger des règles propres à déterminer les paquets autorisés et les paquets interdits ; chaque paquet est caractérisé par quelques paramètres :

- l'interface réseau sur laquelle le paquet est arrivé ; un pare-feu a au moins deux interfaces, l'une connectée au réseau privée et l'autre connectée au lien d'accès à l'Internet ;
- le fait que le paquet se présente sur l'interface depuis l'intérieur du pare-feu ou depuis le réseau ;
- le protocole auquel appartient le paquet, tel que mentionné dans son en-tête IP ;
- les adresses d'origine et de destination, mentionnées dans l'en-tête IP du paquet ;

- les numéros de port d'origine et de destination, mentionnés dans l'en-tête TCP ou UDP ;
- s'il s'agit d'un paquet TCP, les numéros de séquence et d'acquittement, qui permettent de reconstituer la séquence des paquets d'une connexion TCP.

Ces paramètres permettent d'identifier le type de communication auquel appartient le paquet, et éventuellement de reconstituer une séquence. Le simple filtrage par port se traduit par la rédaction de règles simples, qui peuvent prendre la forme de listes de contrôle d'accès (ACL) comme sur les routeurs *Cisco*.

Le logiciel libre *IP Tables / Netfilter*[4], qui permet de construire un pare-feu avec un système Linux, et qui est au cœur de beaucoup de pare-feu du commerce, offre de grandes possibilités en termes de suivi de connexion et d'analyse des paquets. *IP Tables* peut garder des datagrammes en file d'attente pour en reconstituer une séquence complète et en faire l'analyse, ce qui est indispensable si l'on veut détecter des protocoles furtifs comme certains systèmes poste à poste, tels Skype et KaZaA, que nous évoquerons à la page 210.

Routeur filtrant et pare-feu, configurés pour repousser certaines attaques en rejetant des paquets appartenant à des connexions suspectes, produisent des fichiers de comptes rendus (dits journaux, ou *logs*) qui relatent les incidents. Il est bien sûr indispensable, pour bénéficier de la protection qu'ils sont censés procurer, d'analyser le contenu de ces fichiers et de les confronter avec les avis publiés par les CERT *(Computer Emergency Response Teams)* — sur les CERT voir la page 17. Ceci suppose des ingénieurs compétents pour ce faire, ce qu'oublient certaines entreprises qui se croient protégées en achetant (fort cher) un pare-feu clés en mains, configuré avec des filtres pertinents à un instant donné, mais périmés quinze jours plus tard, et qui se retrouvent ainsi dotés d'une magnifique ligne Maginot.

Il existe un marché assez actif du pare-feu, ainsi que des logiciels libres, tel *IP Tables* mentionné ci-dessus, qui permettent de réaliser un pare-feu avec un ordinateur sous un Unix libre. Il existe des logiciels pare-feu pour ordinateur individuel, tel celui que Microsoft incorpore désormais à son système *Windows*, ou *Zone Alarm*. *IP Tables* peut être configuré en pare-feu personnel pour machine Linux isolée ; c'est assez complexe, mais il existe des enrobages pré-configurés très faciles d'emploi, comme par exemple *Firestarter*[5]. Les routeurs ADSL pour réseau

[4] `http://www.netfilter.org/`
[5] `http://www.fs-security.com/`

personnel comportent tous un pare-feu, parfois assez puissant. Tous ces systèmes sont utiles dès lors qu'ils sont correctement paramétrés, mais aucun n'est une panacée. Pour un réseau d'entreprise, l'efficacité d'un pare-feu est subordonnée aux conditions suivantes :

- il y a un ingénieur compétent chargé du pare-feu, qui a le temps de le configurer et d'en analyser les journaux ;
- le pare-feu est du modèle que connaît bien l'ingénieur ;
- le logiciel ou le *firmware* du pare-feu ont reçu les dernières mises à jour de sécurité ;
- il existe un document qui définit ce qui est autorisé sur le réseau ;
- tout ce qui n'est pas explicitement autorisé par le document évoqué ci-dessus est interdit, et cette interdiction est traduite dans les règles du pare-feu.

Les exigences pour un pare-feu personnel sont moins lourdes, parce que le problème est plus simple : en général, il y a une seule adresse IP publique, éventuellement partagée au moyen d'un routeur qui fait la traduction d'adresses (cf. page 147), et il n'y a aucune raison d'autoriser les connexions entrantes, sauf peut-être un accès distant par SSH *(Secure Shell)* ou L2TP *(Layer Two Tunneling Protocol)*. Sur la machine Linux qui sert à rédiger le présent ouvrage, l'auteur utilise le logiciel libre *IP Tables* / *Netfilter*, muni de l'interface *Shorewall*[6] qui en facilite l'emploi. Voici le fichier de règles `/etc/shorewall/rules` :

```
#ACTION  SOURCE  DEST    PROTO   DEST    SOURCE  ORIGINAL
#                                PORT    PORT(S) DEST
ACCEPT    net     fw      icmp    8
ACCEPT    fw      net     icmp
AllowSSH  net     fw
#LAST LINE -- ADD YOUR ENTRIES BEFORE THIS ONE --
```

et le fichier de « *policy* » `/etc/shorewall/policy`, des plus simples :

```
#SOURCE         DEST            POLICY          LOG LEVEL
fw              net             ACCEPT
net             all             DROP            info
# The FOLLOWING POLICY MUST BE LAST
all             all             REJECT          info
#LAST LINE -- ADD YOUR ENTRIES ABOVE THIS LINE --
```

[6] http://www.shorewall.net/

Lors de l'examen d'une demande d'accès réseau, le pare-feu examine d'abord le fichier de règles puis, si aucune règle ne correspond à la situation, le fichier de « *policy* » ; la première ligne de l'un des deux fichiers qui correspond à la situation s'applique, les suivantes ne sont pas examinées. Comme la machine est unique, elle est confondue avec le pare-feu et constitue à elle toute seule la zone `fw`. Les règles disent que l'on autorise les accès au réseau depuis la zone interne `fw` (sortants), ainsi que le protocole ICMP (essentiellement `ping`) du réseau vers la machine et vice-versa, ainsi que l'accès par SSH depuis le réseau. Les autres types d'accès depuis le réseau (entrants) sont proscrits.

J'ai vu des configurations de pare-feu qui se résumaient en une litanie d'ouverture de ports notoirement dangereux pour des adresses IP spécifiques : une telle configuration assure une sécurité nulle, parce que l'usurpation d'adresse IP est un sport que tous les pirates pratiquent depuis l'école maternelle, tellement c'est facile.

Important

On ne le répétera jamais assez, il est indispensable, sur un accès à l'Internet, d'interdire les protocoles de partage de fichiers, notamment Netbios (ports 137, 138, 139, protocoles TCP et UDP).

Protection d'un poste Linux isolé avec Netfilter

La mise en œuvre de *Netfilter* dans un système Linux permet plusieurs usages. Le premier qui vient à l'esprit est de remplacer un boîtier du commerce par un ordinateur Linux disposant de plusieurs cartes Ethernet pour obtenir un pare-feu plus économique. Cette notion d'économie est cependant toute relative : l'utilisateur devra s'assurer que les fonctions offertes sont suffisantes, et qu'à l'arrivée le coût d'exploitation reste compatible avec ses objectifs. D'expérience, l'administration d'un Netfilter nécessite des connaissances techniques plus importantes qu'avec certains produits sur étagère du marché. Que le lecteur ne se prenne cependant pas à croire qu'administrer un pare-feu du marché, aussi bien soit-il, puisse se faire sans compétences.

Un autre usage où *Netfilter* apporte un intérêt certain, c'est la protection du poste de travail Linux isolé : que je sois un particulier raccordé à l'Internet sur une liaison ADSL ou une petite entreprise avec un serveur isolé rendant certains services,

Netfilter me permet de mettre en œuvre un bon niveau de sécurité sans devoir faire l'acquisition d'un boîtier complémentaire.

L'important est de bien définir sa politique de sécurité en fonction de l'usage qui est prévu. La suite de cet exposé proposera une mise en œuvre pour un poste de travail isolé et raccordé de façon permanente à l'Internet tout en disposant d'une adresse IP publique (il n'y a donc pas de traduction d'adresses).

En premier lieu il convient de définir en des termes simples la politique de sécurité qui doit être mise en oeuvre. Celle de notre exemple est très simple :

- la machine Linux héberge un serveur Web (HTTP et HTTPS) qui doit être accessible depuis tout l'Internet ;
- un serveur de courrier (SMTP) reçoit les correspondances de l'extérieur ;
- un accès par le protocole SSH est autorisé à certaines adresses IP en provenance de l'Internet ;
- l'utilisateur local peut utiliser les services HTTP, HTTPS, FTP et SSH depuis la machine vers toute destination de l'Internet ;
- les services DNS et NTP (synchronisation de l'heure) sont autorisés car ce sont des services de base indispensables ;
- tout autre trafic est interdit, mais le trafic ICMP associé aux connexions légitimes doit, lui, être acheminé à destination ;
- le trafic provenant de certaines adresses IP (et notamment des adresses IP privées définies dans la RFC 1918) est interdit, car il n'a aucune raison d'arriver sur la machine en provenance de l'Internet.

L'absence d'une configuration de type « mode diode » est voulue. L'administrateur pourra à loisir faire évoluer la liste des règles pour autoriser des services supplémentaires. Il veillera cependant à conserver lisibilité et simplicité dans les règles et s'interdira d'ouvrir des services dangereux.

Par convention, dans l'exemple de règles ci-après, l'unique interface réseau de la machine Linux s'appelle `eth0`. Les commandes `iptables` proposées sont appelées avant le démarrage de l'interface réseau afin de garantir une protection dès le premier instant. Sur un système Linux-Debian l'administrateur pourra judicieusement placer ces commandes dans un *script* situé dans le répertoire `/etc/network/if-pre-up.d` et l'appeler `iptables`.

```
##! /bin/sh
##
## Ce script est à lancer juste avant la mise à disposition de l'interface
## réseau. Le premier test ci-après n'a de sens que dans un environnement
## Linux Debian lorsque le script est placé dans le répertoire
## /etc/network/if-pre-up.d (ici il s'agit d'activer les règles pour eth0
## et de ne rien faire dans tous les autres cas).
test "${IFACE}" = "eth0" || exit 0

## Charger les modules (si cela n'a pas déjà été fait au
## démarrage du système).
modprobe iptables ip_conntrack ip_conntrack_ftp

## Effacer toutes les règles actuellement présentes sur la machine.
iptables -F; iptables -X

## La première règle vise à interdire tout le trafic dans les principales
## chaînes du module Netfilter. La chaîne INPUT prend en charge les
## paquets à destination de la machine, la chaîne OUTPUT traite les
## paquets émis par la machine et enfin la chaîne FORWARD n'est utilisée
## que si le système assure une fontion de routeur et dispose de plusieurs
## cartes réseau, ce qui n'est pas le cas de notre exemple. Les ordres
## ci-après ont donc pour effet de "fermer" le système : tout le trafic
## réseau est interdit.
iptables -P INPUT   DROP
iptables -P OUTPUT  DROP
iptables -P FORWARD DROP

## Nous allons maintenant définir un certain nombre de chaînes de contrôle
## qui seront utilisées ultérieurement. Ceci contribue à la lisibilité de
## la politique de sécurité.

## listenoire : il s'agit des réseaux IP desquels aucun trafic provenant
## de l'Internet n'est légitime. Cela comprend les adresses privées, mais
## aussi l'adresse IP de la machine et de certains réseaux spécifiques.
iptables -N listenoire
iptables -A listenoire -s 127.0.0.0/8    -j DROP  ## Loopback
iptables -A listenoire -s 10.0.0.0/8     -j DROP  ## RFC-1918
iptables -A listenoire -s 172.16.0.0/12  -j DROP  ## RFC-1918
iptables -A listenoire -s 192.168.0.0/16 -j DROP  ## RFC-1918
iptables -A listenoire -s x.x.x.x        -j DROP  ## Mon adresse IP
iptables -A listenoire -s x.x.x.x/yy     -j DROP  ## Réseau indésirable

## serveurs : les règles à appliquer au trafic en entrée vers les
## quelques services accessibles de l'extérieur (SMTP, HTTP, HTTPS
## & SSH depuis certains réseaux seulement).
iptables -N serveurs

## Autoriser le courrier électronique (port 25, SMTP)
iptables -A serveurs -p TCP --dport smtp \
  -m state --state NEW,ESTABLISHED -j ACCEPT
```

```
## Les accès aux serveurs WWW
iptables -A serveurs -p TCP --dport http \
  -m state --state NEW,ESTABLISHED -j ACCEPT
iptables -A serveurs -p TCP --dport https \
  -m state --state NEW,ESTABLISHED -j ACCEPT
## Le réseau 194.2.11.128/25 peut se connecter avec SSH
iptables -A serveurs -p TCP --dport ssh -s 194.2.11.128/25 \
  -m state --state NEW,ESTABLISHED -j ACCEPT

## autres_in : règles pour attraper tout ce qui doit l'être, c'est-à-dire
## le trafic retour des connexions TCP et UDP établies (à l'initiative de
## la machine) mais aussi le trafic ICMP en rapport avec des connexions
## établies.
iptables -N autres_in
iptables -A autres_in -p TCP \
  -m state --state ESTABLISHED,RELATED -j ACCEPT
iptables -A autres_in -p UDP \
  -m state --state ESTABLISHED         -j ACCEPT
iptables -A autres_in -p ICMP \
  -m state --state RELATED             -j ACCEPT

## Appliquer les règles de sécurité sur le trafic provenant de l'Internet
## et destiné à la machine Linux.
iptables -A INPUT  -i eth0 -j listenoire    ## La liste noire
iptables -A INPUT  -i eth0 -j serveurs      ## Le serveurs sur ma machine
iptables -A INPUT  -i eth0 -j autres_in     ## Autres sessions
iptables -A INPUT  -i eth0 -j DROP          ## Tout le reste à la poubelle

## clients : règle pour autoriser le trafic en sortie, ce que ma machine a
## le droit de faire en direction de l'Internet.

## Le service DNS (ports 53 UDP & TCP)
iptables -N clients -p TCP --dport domain \
  -m state --state NEW,ESTABLISHED    -j ACCEPT
iptables -N clients -p UDP --dport domain \
  -m state --state NEW,ESTABLISHED    -j ACCEPT
## File Transfer Protocol
iptables -N clients -p TCP --dport ftp \
  -m state --state NEW,ESTABLISHED    -j ACCEPT
## Synchronisation de l'heure
iptables -N clients -p UDP --dport ntp \
  -m state --state NEW,ESTABLISHED    -j ACCEPT
## La navigation "standard" vers l'Internet
iptables -N clients -p TCP --dport http \
  -m state --state NEW,ESTABLISHED    -j ACCEPT
iptables -N clients -p TCP --dport https \
  -m state --state NEW,ESTABLISHED    -j ACCEPT
## Envoyer du courrier électronique
iptables -N clients -p TCP --dport smtp \
  -m state --state NEW,ESTABLISHED    -j ACCEPT
## Autoriser toute connexion SSH sortante
```

```
iptables -N clients -p TCP --dport ssh \
  -m state --state NEW,ESTABLISHED    -j ACCEPT

## autres_out : règles pour attraper tout ce qui doit l'être, c'est-à-dire
## le trafic retour des connexions TCP et UDP établies (à l'initiative de
## l'extérieur comme par exemple une connexion SMTP entrante).
iptables -N autres_out
iptables -A autres_out -p TCP \
  -m state --state ESTABLISHED,RELATED -j ACCEPT
iptables -A autres_out -p UDP \
  -m state --state ESTABLISHED         -j ACCEPT
iptables -A autres_out -p ICMP \
  -m state --state RELATED             -j ACCEPT

## Appliquer les règles de sécurité sur le trafic sortant (c'est à dire
## celui de la machine à destination de l'Internet).
iptables -A OUTPUT -o eth0 -j clients    ## Les services "client" autorisés
iptables -A OUTPUT -o eth0 -j autres_out ## Autres sessions
iptables -A OUTPUT -o eth0 -j DROP       ## Tout le reste à la poubelle
```

Important

Les règles proposées utilisent le module *connection tracking* du noyau Linux (ce module fait partie intégrante de *Netfilter*). Son intérêt est de faire évoluer *Netfilter* d'un simple filtre de paquets (pour une liste de contrôle d'accès sur un routeur IP) vers un pare-feu à états *(stateful firewall)*. On ne saurait aujourd'hui demander moins, les filtres de paquets étant insuffisants dans la quasi-totalité des situations qui peuvent se présenter.

Listes de contrôle d'accès pour le réseau

Les listes de contrôle d'accès *(access control list, ACL)* pour contrôler les accès à un réseau, introduites par *Cisco*, utilisent les mêmes principes que les ACL Posix évoquées à la page 44, mais elles sont adaptées, plutôt qu'aux fichiers, aux objets du réseau : interfaces réseau, adresses IP et ports. Ainsi, les entrées ci-dessous :

```
int Ethernet 1
access-group 101 in
int serial 0
access-group 101 in
```

disent que le filtrage des paquets en entrée du routeur (paramètre `in`) sur les interfaces `Ethernet 1` et `serial 0` sera conforme aux règles contenues dans la liste d'accès n° 101, que voici :

```
access-list 101 permit tcp any host 192.168.35.1 eq www
access-list 101 permit tcp any host 192.168.35.1 eq 443
access-list 101 permit icmp any host 192.168.35.1 eq echo
```

Ces règles disent que le serveur à l'adresse `192.168.35.1` ne peut être atteint que par les deux ports Web ouverts (`www` est un raccourci pour le port 80), et qu'il peut aussi être atteint par la commande `ping`.

Les pare-feu personnels pour ordinateurs sous Windows

Un pare-feu se présente habituellement comme un appareil qui s'installe en rupture de flux entre plusieurs zones de sécurité : une situation courante est une séparation entre trois espaces que sont l'Internet (réseau dangereux par nature), une DMZ (qu'il est important de protéger mais qui peut être amenée à discuter avec l'Internet) et le réseau interne qui ne doit recevoir aucune connexion de l'extérieur mais peut être amené à communiquer de façon contrôlée avec la DMZ et parfois même l'Internet.

La logique de conception d'une politique de sécurité repose dans ce cas sur la définition de règles de flux. Les connexions autorisées sont définies à partir de l'adresse IP source (éventuellement d'un numéro de port) et d'un protocole, cela à destination d'un service bien particulier (numéro de port) et peut-être même d'une plage restreinte d'adresses IP. Les produits classiques (dits « à états », ou *stateful*) vérifieront la conformité du trafic au niveau du protocole de couche 4 (TCP ou UDP) alors que des produits plus évolués (et souvent plus coûteux) pourront en plus s'assurer d'une certaine conformité au protocole supposé être transporté.

En complément (ou parfois au titre de seule et unique protection pour un poste isolé) de ces produits de sécurité situés aux extrémités du réseau il est de plus en plus courant d'équiper ses micro-ordinateurs d'un logiciel pare-feu. L'utilisateur d'un système de type *Linux* pourra faire cela à moindres frais : *Netfilter* lui permettra de protéger son poste de travail tout en concevant ses règles de sécurité de la même façon qu'il le ferait pour un pare-feu situé dans le réseau (cf. la section 128). L'utilisateur d'un système *Windows* récent n'est pas oublié non plus : que ce soit

le pare-feu intégré au système d'exploitation *Windows XP* ou bien un produit du commerce, comme le pare-feu *Sygate* ou l'un de ses nombreux dérivés, l'utilisateur peut mettre en œuvre des moyens de protection de son ordinateur.

La philosophie de fonctionnement et de gestion, voire de conception, des règles de sécurité est cependant très différente entre ces produits et les plus classiques pare-feu réseau : si ces outils permettent, avec plus ou moins de facilité, la définition de règles réseau, leur intérêt principal est d'y associer des autorisations d'accès au réseau pour des programmes. Une règle de sécurité autorisant le courrier électronique pourrait ainsi devenir « j'autorise le programme *Thunderbird*, et lui seul, à se connecter sur le port 25 (SMTP), afin d'envoyer du courrier électronique ».

Dans la pratique, les interfaces de gestion proposées ne sont pas aussi pratiques qu'on le souhaiterait : dans un souci de simplicité (car l'utilisateur n'est en général pas un expert de sécurité informatique), les autorisations d'accès au réseau que l'utilisateur peut être amené à donner restent assez larges.

Un mode de fonctionnement assez typique de ce genre de produit est le suivant :

- à l'installation un certain nombre de règles par défaut autorisent du trafic sur le réseau (certains programme du système) ;
- lorsqu'un nouveau programme souhaite accéder au réseau, le pare-feu personnel détecte cette action et va émettre une alerte auprès de l'utilisateur, lui laissant le choix : autoriser ou bloquer ce trafic (et éventuellement se souvenir de ce choix pour la prochaine fois) ;
- à moins d'utiliser des règles avancées, pas toujours faciles à configurer, l'autorisation précédemment donnée vaut pour tout le trafic réseau que le programme pourrait être amené à traiter, qu'il soit pour le service ayant généré l'alerte ou bien pour un autre service - et c'est là qu'est le danger.

Le pare-feu du système d'exploitation Windows XP

Lorsque l'option pare-feu est activée dans Windows XP (cela se fait dans les propriétés avancées d'une carte réseau), elle permet de mettre l'ordinateur dans un mode de fonctionnement de type « diode » : tout le trafic sortant est autorisé, mais le trafic entrant est, lui, interdit, sauf exception. Ces exceptions peuvent se configurer manuellement ou bien donner lieu à une alerte lorsqu'elles se présentent. Illustrons par deux exemples.

- **Situation n° 1 :** connexion SSH vers une machine distante en utilisant le programme *Putty*. Cette situation ne donnera lieu à aucune alerte puisque, s'agissant d'une connexion sortante, elle n'est pas bloquée par le pare-feu Windows XP.
- **Situation n° 2 :** l'utilisateur lance le programme VLC (Vidéolan) et y configure un serveur pour diffuser du contenu sur le port TCP n° 1234. Cette action sera interceptée par le pare-feu Windows XP dès que le programme VLC indique au système qu'il est prêt à recevoir les connexions TCP sur le port n° 1234. Le pare-feu ouvrira alors une alerte pour demander à l'utilisateur de décider (figure 6.9).

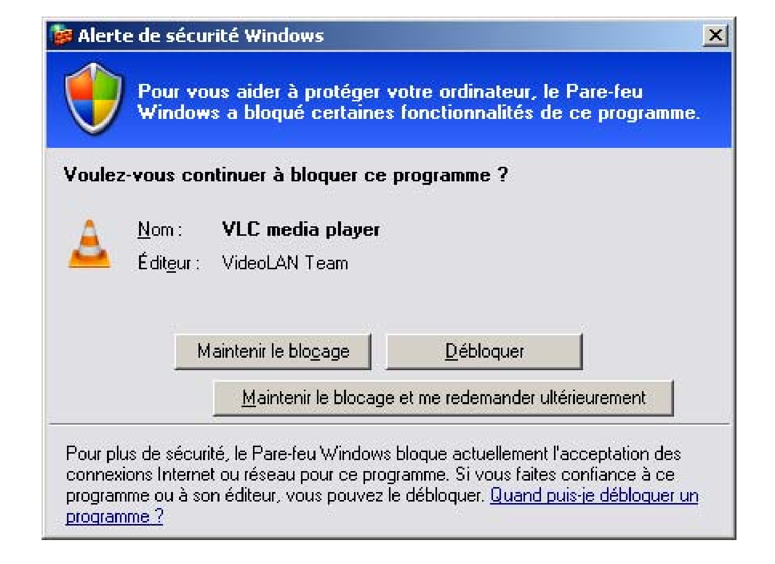


Figure 6.9
Panneau d'alerte du pare-feu de *Windows*

Cette alerte, simple à comprendre, laisse cependant l'utilisateur expert sur sa faim : elle ne donne aucune information quant à la nature du trafic réseau qu'il faut autoriser (ou interdire), probablement parce que la décision par défaut sera, en cas de déblocage, d'autoriser tout trafic entrant destiné au programme VLC, que ce soit sur le port n° 1234 ou bien pour un autre service.

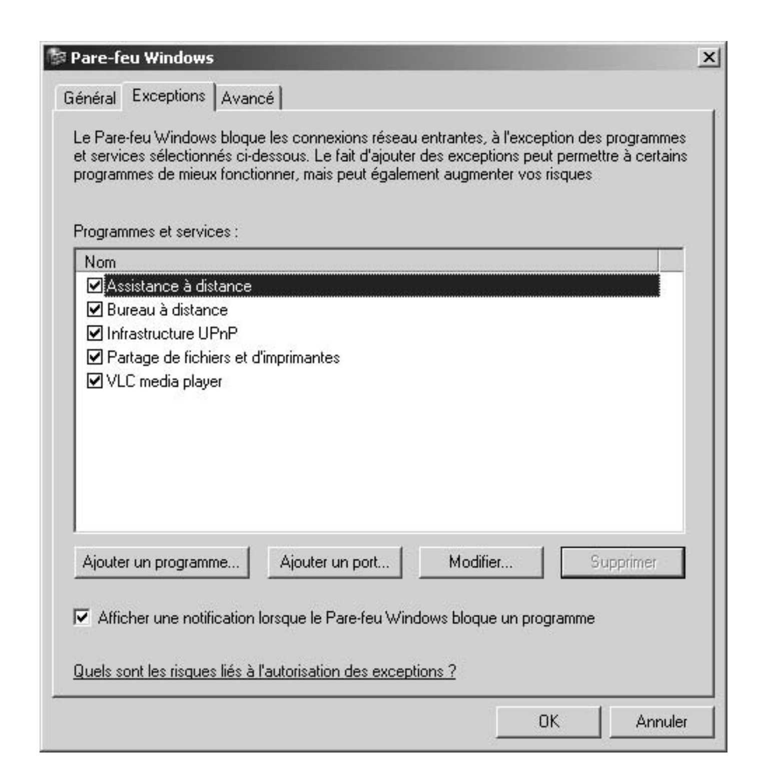


Figure 6.10
Panneau de configuration du pare-feu de *Windows*

Il est bien sûr possible de consulter à tout moment l'état des autorisations (via les propriétés avancées de la carte réseau), comme on le voit sur la figure 6.10.

Ce tableau est d'ailleurs l'occasion de voir l'existence de quelques exceptions « par défaut » qu'il n'est pas possible de supprimer (les quatre premières lignes de la copie d'écran ci-dessus).

C'est également depuis cet écran que l'utilisateur pourra modifier les règles, par exemple pour restreindre l'autorisation de base accordée par le pare-feu Windows et limiter les plages d'adresses IP ou les numéros de ports autorisés.

Un pare-feu personnel dérivé de la technologie Sygate

Lorsque le système n'intègre pas de pare-feu en standard (par exemple Windows 2000) ou que les fonctions du pare-feu Windows XP sont jugées insuffisantes, il existe une pléthore de produits qui pour quelques dizaines d'euros permettent de rendre des services de sécurité. Nous illustrons cette section avec des copies d'écran issues du produit *Netscreen Remote Security Client* (version datant de 2002), qui utilise la technologie Sygate.

Le principe de fonctionnement est similaire au produit précédent, mais il intègre quelques fonctions complémentaires qui peuvent être utiles, par exemple :

- la détection de certaines attaques comme un *port scan* ;
- la possibilité d'autoriser ou d'interdire les services réseau Microsoft (partage d'imprimante, de fichiers, ou utilisation de tels serveurs) ;
- des facilités pour mettre en œuvre des règles « réseau » comme avec un pare-feu classique.

Le système d'autorisation, programme par programme reste comparable à celui du pare-feu Windows XP, avec en plus un contrôle aussi bien en entrée qu'en sortie.

Reprenons l'un des exemples précédents pour illustrer les quelques différences : connexion SSH vers la machine `lamachinede.mondomaineamoi.fr` en utilisant le programme *Putty*. Le pare-feu émettra l'alerte illustrée par la figure 6.11 même pour cette connexion sortante, parce que ce programme n'a pas été autorisé lors d'une utilisation précédente.



Figure 6.11
Mire d'alerte du pare-feu *Sygate*

Ici aussi, il faudra, lors d'une deuxième opération, utiliser les propriétés avancées dans la liste des programmes autorisés pour préciser le souhait de n'autoriser que les connexions sortantes TCP sur le port n° 22, comme indiqué sur la figure 6.12.

Si cette configuration avancée est relativement simple à faire pour un programme comme `putty`, qu'en est-il lorsqu'il s'agit d'indiquer au pare-feu des règles avancées pour un programme système, comme celui montré sur la figure 6.13 ?

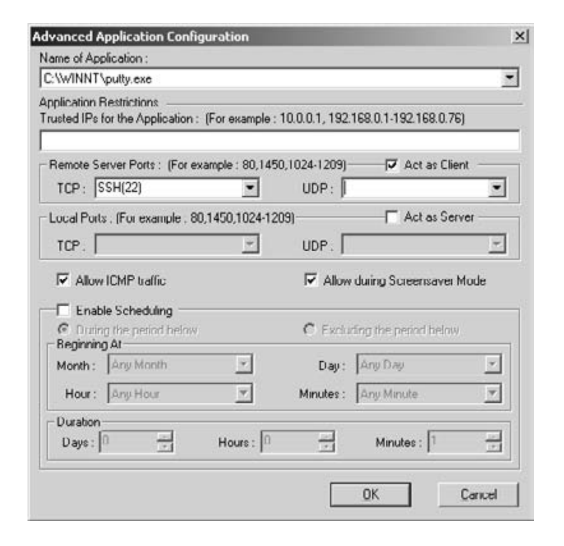


Figure 6.12
Panneau de configuration du pare-feu *Sygate*



Figure 6.13
Panneau de configuration du pare-feu *Sygate*

À moins d'être un expert du fonctionnement de Windows 2000, bien peu de personnes sont en mesure de dire quels sont les flux qui peuvent être autorisés pour le programme `inetinfo.exe`, et dans ce cas on se contentera du choix par défaut, à savoir : tout trafic réseau de ce programme est autorisé, sans restrictions.

Le système de noms de domaines (DNS)

Le DNS, ou système de noms de domaines *(Domain Name System)* est le système d'annuaire réparti destiné à traduire, sur l'Internet, des noms de serveurs en adresses réseau.

Pour établir par un réseau d'ordinateurs une liaison selon le protocole IP *(Internet Protocol)*, le nœud à l'initiative de la communication doit émettre des paquets de données vers le nœud destinataire, qui lui répondra. Pour ce faire, les paquets doivent comporter un en-tête qui comprendra notamment les adresses d'origine (le nœud émetteur) et de destination (le nœud destinataire). Mais un utilisateur de l'Internet ne connaît généralement pas les adresses des machines avec lesquelles il veut communiquer. Ce qu'il veut faire, le plus souvent, c'est envoyer un courrier électronique, par exemple à l'INRIA, ou consulter le serveur `http://www.sncf.fr` pour connaître l'horaire de train. `www.sncf.fr` n'est pas une adresse, mais un nom qui désigne la machine qui abrite le serveur désiré. `sncf.fr` n'est pas une adresse mais un nom qui désigne un domaine au sein duquel se trouve par exemple un serveur de courrier électronique nommé `mail.sncf.fr`, qui détient la boîte aux lettres électronique d'un certain nombre d'abonnés. Mais la couche réseau IP n'a que faire des noms, elle ne connaît que des adresses.

Fonctionnement du DNS

Il faut se dire qu'avant même de résoudre cette embarrassante affaire de noms et d'adresses, ce que l'on veut envoyer ce ne sont pas des paquets IP, mais des courriers électroniques ou des interrogations au serveur. Mais là, la réponse est aisée. Tout bon logiciel de courrier électronique donnera à notre message la mise en forme convenable (définie par une *Request for Comment* (RFC) fameuse entre tous, la RFC 822, mise au goût du jour par le 2822), puis le transmettra à un logiciel serveur de messagerie (couche application) conforme au protocole de transport de courrier électronique SMTP *(Simple Mail Transfer Protocol)* tel que *Sendmail* ou *Postfix*, qui s'occupera de déterminer comment atteindre le destinataire, et transférera toutes les informations et données nécessaires au protocole de transport TCP (couche 4 de l'OSI), qui lui-même entamera les manœuvres nécessaires en envoyant des flux de bits à la couche IP (couche 3 de l'OSI), qui découpera tout cela en paquets avec les bonnes données et les bonnes adresses, et les enverra à la couche liaison de données (couche 2 de l'OSI).

Les passerelles de messagerie

Sendmail et *Postfix* sont des logiciels dits de *passerelle de messagerie (Mail Transfert Agent, MTA)*, de même que *Qmail* et *Exim*. Une passerelle de messagerie est chargée de l'expédition et de la distribution du courrier électronique, à la différence de logiciels comme *Eudora*, *Thunderbird* ou *Outlook*, qui sont des *clients de messagerie (User Agent, UA)*, chargés de remettre le courrier départ au MTA et de relever la boîte à lettres. *Sendmail*, conçu par Eric Allman, a joué un rôle historique considérable dans le développement de l'Internet, c'est l'exemple de ces logiciels libres sans lesquels l'Internet n'existerait pas. Postfix, conçu par Wietse Venema, est réputé pour sa sécurité.

Si l'on revient maintenant à la question initiale, le nom d'un serveur (de courrier électronique, Web, etc.) est connu et ce qu'il faut à la couche IP c'est son adresse, qui ici joue plutôt le rôle du numéro de téléphone. Dans la vie courante, pour répondre à ce genre de question il y a des annuaires : sur l'Internet c'est la même chose. L'annuaire qui permet, si l'on connaît le nom d'un serveur, de trouver son adresse, et vice-versa, s'appelle le DNS (Domain Name System).

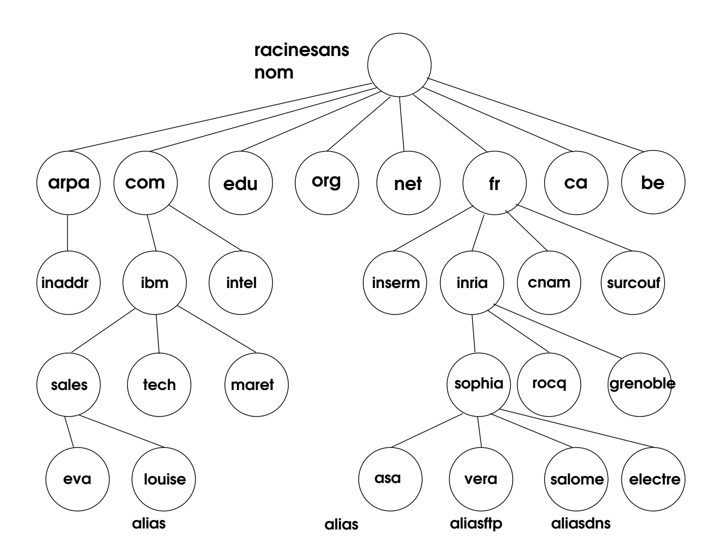
Fonctionnellement, la description du DNS tient en deux lignes : c'est une liste à deux colonnes, dans la colonne de droite on a les noms, dans la colonne de gauche les adresses. D'ailleurs aux origines de l'Internet il en était réellement ainsi, et les fichiers `hosts` en conservent le souvenir. Aujourd'hui le DNS est une immense base de données distribuée sur l'ensemble de la planète, l'une des plus grandes qui existent. Le processus de résolution de noms en adresses est complété par un autre service, qui publie les noms des serveurs de courrier électronique qui desservent un domaine et permet ainsi la distribution planétaire des messages électroniques. Il faut définir maintenant ce qu'est un domaine, et ceci fait l'objet de l'alinéa suivant.

Un espace abstrait de noms de serveurs et de domaines

L'espace des noms de l'Internet (il est important de garder à l'esprit que les schémas qui vont suivre décrivent un espace abstrait de noms de serveurs et de domaines, c'est-à-dire la structure d'un ensemble d'informations, et pas la topologie du réseau physique qui les relie), est organisé de façon hiérarchique, selon un schéma calqué sur l'organisation du système de fichiers Unix. Ce système génial, dont le fonctionnement n'est pas très facile à comprendre, a été défini par Paul Mockapetris dans les RFC 1034 et 1035.

La figure 6.14 montre l'organisation hiérarchique de l'espace de noms de l'Internet. Chaque nœud de l'arbre, représenté par un cercle, comprend un label, qui peut avoir jusqu'à 63 caractères de long, et pour lequel les lettres minuscules et majuscules ne sont pas distinguées. Le nom de domaine d'un nœud s'obtient en construisant la séquence de tous les labels des nœuds compris entre le nœud considéré et la racine inclus, séparés par des points, par exemple `vera.sophia.inria.fr`.

Figure 6.14
Organisation en arbre des noms de domaines



Sous une racine sans nom se trouvent un certain nombre de domaines de premier niveau (TLD, pour *Top Level Domains*). Chaque entreprise, association, université ou autre entité désireuse d'accéder à l'Internet appartiendra à un de ces domaines. Ceux qui ont des noms à trois lettres sont dits domaines génériques : `com`, `edu`, `net`, `gov`, respectivement pour les activités commerciales, éducatives, liées au réseau ou rattachées au gouvernement américain. Les TLD à deux lettres sont des domaines géographiques : `fr`, `ca`, `be`, `de`, `dz` respectivement pour la France, le Canada, la Belgique, l'Allemagne et l'Algérie. Le domaine `arpa` a un rôle particulier, il sert à la résolution inverse, c'est-à-dire à la traduction des adresses en noms.

Les serveurs DNS racine de l'Internet

Les *serveurs racine* du DNS utilisé dans l'Internet et connus sous les noms `a.root-servers.net` à `m.root-servers.net` sont représentés sur l'Internet par 13 adresses IP. Dans la réalité le nombre d'équipements est bien sûr très supérieur à 13.

Certains serveurs racine sont constitués de plusieurs machines (éventuellement derrière un répartiteur de charge) afin d'assurer des redondances et permettre le traitement d'un nombre de requêtes plus important.

D'autres (par exemple `K`) sont déployés en utilisant une technique fort intéressante pour une telle application : le routage *Anycast*. Cette technique consiste à répliquer l'adresse IP en de nombreux lieux géographiques judicieusement choisis. Le protocole de routage de l'Internet *(Border gateway protocol, BGP)* fera le reste du travail et acheminera les utilisateurs auprès du serveur `K` (disponible) le plus proche.

Une autre mesure complémentaire visant à augmenter le niveau de sécurité consiste en l'utilisation de matériels, systèmes d'exploitation et logiciels différents pour rendre la même fonction : ainsi un défaut qui affecterait une technologie précise n'aura peut-être pas d'impact sur un produit différent.

Pour en savoir plus sur les serveurs racine, le site `http://www.root-servers.org/` peut servir de point d'entrée.

Autres niveaux de domaines

Au niveau inférieur, au sein du TLD, on trouve généralement les domaines qui correspondent aux universités, aux entreprises, etc. qui se sont connectées à l'Internet. Elles ont choisi elles-mêmes leur nom de domaine, avec la contrainte que le nom complet doit être unique : il ne peut y avoir qu'un domaine `inria.fr`, mais il peut y avoir `ibm.com`, `ibm.fr`, `ibm.be`, etc. Ces domaines peuvent être eux-mêmes subdivisés : ainsi l'INRIA (Institut National de la Recherche en Informatique et en Automatique) aura un domaine pour chacune de ses unités de recherche, Rocquencourt, Sophia-Antipolis, Grenoble, etc., qui s'appelleront `sophia.inria.fr`, `rocq.inria.fr`, `grenoble.inria.fr`, etc.

Cette subdivision peut atteindre des niveaux plus ou moins fins ; les feuilles de l'arbre, au niveau le plus bas, correspondent aux nœuds du réseau, qui sont des stations de travail ou des serveurs.

Une station sur le réseau peut avoir, outre son nom propre tel que nous venons de le voir, un ou plusieurs alias. Ainsi il est de coutume que le serveur Web d'un organisme soit connu sous le nom `www.quelquechose.fr`. Alors si le serveur Web

de l'INRIA Sophia est hébergé sur la machine `asja`, celle-ci recevra un alias, `www.sophia.inria.fr`. Les deux noms désigneront la même machine, ou plus exactement la même interface sur le réseau.

Il serait possible d'imaginer une administration centralisée de l'arbre des domaines, mais une fraction de seconde de réflexion révélerait l'immensité des difficultés qui en résulteraient. Aussi cet arbre est-il découpé en sous-arbres appelés zones, administrées séparément. Ainsi en France l'Association française pour le nommage Internet en coopération (AFNIC) administre-t-elle tous les domaines dont le nom se termine par .fr : on dit que l'AFNIC a reçu délégation pour la zone fr. De même l'AFNIC déléguera l'administration de la zone `inria.fr` à l'INRIA, qui lui-même déléguera à une équipe de son unité de Sophia-Antipolis l'administration de `sophia.inria.fr`.

Dès lors qu'un organisme a reçu délégation de l'administration d'une zone, il a le devoir de mettre en service des serveurs de noms pour cette zone, au moins deux, un primaire et un secondaire (les termes primaire et secondaire tendent à être remplacés par maître et esclave). Un serveur de noms est un logiciel que l'on peut interroger : si on lui fournit le nom d'une machine, il renvoie son adresse. Dès qu'un nouvel ordinateur est mis en service dans une zone, l'administrateur du DNS de cette zone doit lui affecter un nom et une adresse et les ajouter à la base de données du serveur de noms primaire local. On dit que ce serveur de noms détient l'autorité sur la zone, il possède l'attribut *Start of Authority (SOA)*.

Un serveur primaire obtient les informations relatives à sa zone en accédant directement aux bases de données locales. Un serveur secondaire (il peut y en avoir plusieurs, et il est recommandé qu'ils soient physiquement distincts et redondants) obtient ces mêmes informations en les demandant au serveur primaire. L'opération par laquelle un serveur secondaire reçoit du serveur primaire l'information qui décrit la zone est nommée *transfert de zone*. La pratique courante est de demander à une personne sur un autre site d'administrer le serveur secondaire pour votre zone, à charge de revanche.

Conversations entre serveurs de noms

Donc tout système installé dans la zone, lorsqu'il voudra traduire un nom en adresse, posera la question au serveur de la zone. Plus précisément, le logiciel d'application qui a besoin de l'adresse (par exemple votre navigateur Web ou le

logiciel de transfert de courrier électronique) fait appel à un résolveur, qui va dialoguer avec le serveur de noms qui lui aura été désigné.

Si le nom à traduire désigne une machine locale, le serveur de noms interrogera directement sa base. Sinon, il doit interroger un autre serveur de noms, qui, lui, connaîtra la réponse. Comment trouver un serveur de noms en possession de la réponse à la question posée ? Chaque serveur connaît (il en possède les adresses dans sa base de données) la liste des serveurs de noms racine, à ce jour au nombre de treize, dispersés à la surface de la planète mais surtout aux États-Unis qui en abritent dix. Ces serveurs racine détiennent la liste des serveurs de noms qui détiennent l'autorité pour tous les domaines de second niveau (dans notre schéma de la figure 6.14 page 141, la ligne `ibm.com`, `inria.fr`, etc.).

Notre serveur va donc interroger un serveur racine. Celui-ci répondra en donnant l'adresse du serveur qui détient l'information autorisée relative au domaine de second niveau dont relève le nom de domaine que l'on cherche à résoudre ; ce troisième serveur, interrogé, donnera soit la réponse, soit l'adresse d'un quatrième serveur plus proche du domaine concerné, etc. Le serveur interrogé initialement peut transmettre la première réponse au résolveur, à charge pour ce dernier d'interroger le serveur de noms suivant, et ainsi de suite : une telle interrogation est dite *itérative*. Le résolveur peut au contraire demander au serveur de faire son affaire des interrogations des autres serveurs de noms impliqués, et de ne transmettre que la réponse finale : une telle interrogation sera dite *récursive*.

Toute cette subtile conversation entre serveurs sera bien sûr ignorée de l'utilisateur. Les logiciels de courrier électronique ou de navigation sur le Web savent faire appel au résolveur. Lorsqu'un abonné individuel à l'Internet allume son modem, la plupart du temps le routeur de son FAI lui envoie, grâce au protocole DHCP *(Dynamic Host Configuration Protocol)*, en même temps que son adresse IP dynamique, l'adresse du ou des serveurs de noms auxquels le résolveur pourra s'adresser. Mais il est utile de savoir à quoi correspond la case la plus perturbante du menu de configuration d'un accès au réseau : celle où l'on demande l'adresse du serveur DNS. Heureusement les méthodes modernes de gestion de réseau évitent presque toujours d'avoir à la remplir.

Sécurité du DNS

Le DNS, on l'aura compris, est un élément constitutif primordial de l'Internet, et de ce fait la sûreté de son fonctionnement doit être maintenue à tout prix. Nous n'envisagerons pas ici les questions qui se posent au niveau des serveurs racine et de l'infrastructure au cœur du réseau, mais plutôt celles que doit se poser l'administrateur d'un réseau local d'entreprise.

Comme c'est le DNS qui résout pratiquement chaque demande d'accès à l'Internet ou au réseau local en donnant l'adresse qui correspond au nom du service demandé, s'il ne fonctionne plus, l'Internet et le réseau sont inaccessibles. Si les serveurs légitimes sont remplacés par des serveurs frauduleux, ou si les serveurs légitimes sont frauduleusement alimentés en données falsifiées, il sera possible de détourner le trafic au profit des fraudeurs, par exemple pour se procurer des informations confidentielles ou pour commettre des escroqueries. Il existe plusieurs méthodes de corruption du DNS, contre lesquelles il importe de se prémunir.

Espace public et espace privé

Le paradoxe du DNS est qu'il doit rester sûr tout en étant ouvert virtuellement au monde entier. Une des premières précautions que prendra un administrateur de réseau avisé sera de séparer d'une part les informations relatives à son réseau qui sont destinées à être visibles du monde entier, par exemple l'adresse du serveur Web public de l'entreprise et celle de sa passerelle de messagerie, d'autre part les informations qui n'ont aucune raison de sortir du réseau local, comme les adresses des serveurs de fichiers et des imprimantes.

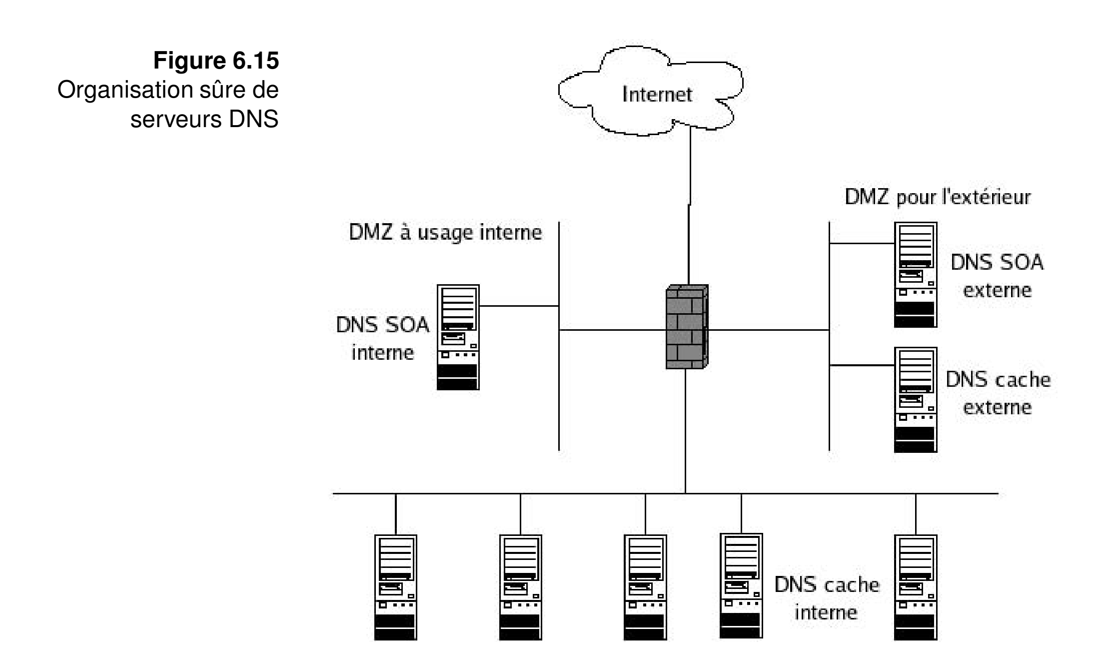
Dans un article consacré à la protection du DNS [24] Christophe Brocas et Jean-Michel Farin analysent les risques liés à son usage et proposent quelques principes pour une solution. Les risques se rangent dans les catégories suivantes :

- fuites d'information par détournement du protocole (ce que l'on appelle les canaux cachés) ;
- obtention indésirable d'informations sur le réseau de l'entreprise au moyen de transferts de zone illégitimes ;
- corruption de la résolution de noms, avec pour conséquence le détournement d'un trafic légitime vers des destinations mises en place à des fins malintentionnées ;

- dénis de service déclenchés par la corruption des bases de données des serveurs DNS.

Afin de se prémunir contre ces risques, les auteurs proposent un certain nombre de mesures qui exploitent les fonctions introduites par les versions relativement récentes des logiciels de gestion du DNS, notamment *Bind*. Ces mesures sont décrites dans le contexte d'un réseau interne d'entreprise, relié à l'Internet au travers d'un pare-feu, et doté de deux DMZ (cf. page 123 pour l'explication) : une DMZ externe où sont installés les serveurs de l'entreprise qui doivent être « vus » depuis l'Internet, une DMZ à usage interne, où résident les serveurs qui doivent « voir » l'Internet. Cette architecture est résumée par la figure 6.15.

Figure 6.15
Organisation sûre de serveurs DNS



La première mesure de protection consiste à interdire aux machines du réseau interne tout accès direct à l'Internet : les accès aux Web sont relayés par un mandataire HTTP, tout le courrier électronique émis doit impérativement être relayé par la passerelle de messagerie de l'entreprise.

La seconde mesure de protection consiste à avoir deux serveurs DNS maîtres, un qui *fait autorité* (et de ce fait nommé *Start of Authority*, SOA) vis-à-vis du monde extérieur, l'autre SOA à usage interne. Le serveur à usage externe divulgue une vue du réseau de l'entreprise réduite aux seules machines qui doivent être visibles de l'Internet. Le serveur à usage interne détient les informations complètes sur le réseau interne.

La troisième mesure de sécurité interdit toute interrogation directe du serveur SOA interne : les stations ne peuvent interroger qu'un serveur cache, esclave du précédent.

La quatrième mesure est la suivante : lorsque le serveur cache interne veut résoudre un nom de domaine inconnu, c'est-à-dire qui correspond à une adresse du monde extérieur, sur l'Internet, il transmet la requête à un serveur cache en DMZ externe.

La cinquième mesure indique que les serveurs SOA ne doivent répondre que pour des requêtes qui concernent un ou plusieurs domaines propres à l'entreprise, ils n'effectuent donc aucune interrogation récursive.

Des mesures complémentaires visent à empêcher les utilisateurs de postes de travail de contourner les mesures de sécurité en réécrivant de façon créative leur fichier `hosts` ; les méthodes d'attribution d'adresses qui exigent la modification dynamique du DNS (DHCP, contrôleur de domaine Windows) demandent des précautions supplémentaires, qui pour bien faire devraient aller jusqu'à l'authentification des transactions.

On mesure, par le résumé succinct qui est donné ici d'un article fort détaillé, que l'administration sûre du DNS n'est pas une affaire facile.

Traduction d'adresses (NAT)

Le système de traduction d'adresses[7] NAT *(Network Address Translation)* est apparu en 1994 dans la RFC 1631 [46] (remplacé maintenant par le 3022 [105]), initialement pour permettre la communication entre l'Internet et des réseaux privés contenant des adresses IP non conformes au plan d'adressage de l'Internet, et il a été ensuite très largement utilisé pour pallier le déficit d'adresses IP engendré par

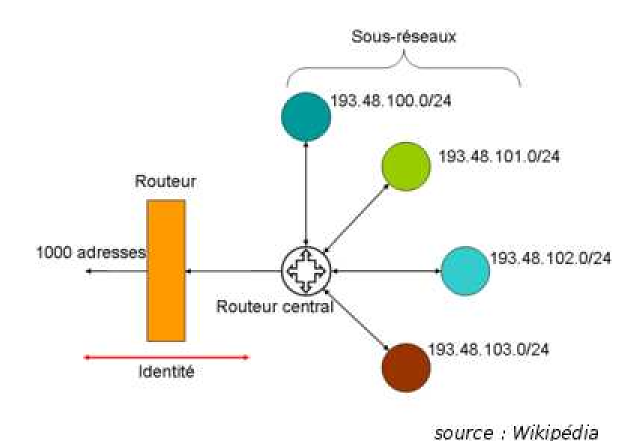
[7] Notons que l'anglais *translation* se traduit ici en français par *traduction*, translation d'adresse ne veut rien dire.

l'étroitesse de la plage d'adresses de la version 4 du protocole. Il est devenu de ce fait à la fois une solution et un problème de sécurité des réseaux.

Le principe du standard téléphonique d'hôtel

Le principe de la traduction d'adresses est le suivant : chaque nœud de l'Internet doit posséder une adresse IP pour mettre en œuvre le protocole TCP/IP, et cette adresse doit être unique, comme pour les numéros de téléphone, sinon il serait impossible d'acheminer correctement les communications.

Figure 6.16
Réseau sans NAT : les adresses des hôtes sont des adresses uniques et routées sur l'Internet.



Mais, pour poursuivre la comparaison avec le téléphone, dans certains hôtels par exemple, seul le standard a un numéro de téléphone unique, et le poste de chaque chambre a un numéro local, à usage strictement interne, et qui peut très bien être le même que celui d'une chambre dans un autre hôtel : cela n'a aucune conséquence fâcheuse car le numéro de la chambre n'est pas visible de l'extérieur ; ceci permet parfaitement à l'occupant de la chambre d'appeler l'extérieur en donnant un code particulier (« composer le numéro 0 pour avoir l'extérieur »), et de recevoir des communications passant par le standard qui effectue la commutation vers la ligne de la chambre.

Adresses non routables

Le système NAT repose sur un principe analogue : dans un réseau local, seuls les nœuds qui ont vocation à abriter des serveurs vus de tout l'Internet, comme le serveur Web de l'entreprise ou sa passerelle de messagerie, doivent recevoir des adresses reconnues universellement, et donc uniques et conformes au plan d'adressage de l'Internet. Les postes de travail ordinaires peuvent recevoir des adresses purement locales, qui ne sont pas routables, c'est-à-dire qu'un paquet à destination d'une telle adresse peut circuler sur le réseau local et atteindre sa destination, mais ne peut pas franchir un routeur, parce que ces classes d'adresses sont explicitement désignées pour que les routeurs les oublient. Sont dites *non routables* toutes les adresses appartenant aux blocs d'adresses définis à cet effet par la RFC 1918 [90] : 192.168.0.0 à 192.168.255.255 (préfixe 192.168/16), 172.16.0.0 à 172.31.255.255 (préfixe 172.16/12) et 10.0.0.0 à 10.255.255.255(préfixe 10/8).

Accéder à l'Internet sans adresse routable

Si la gestion des adresses non routables s'arrêtait là, ces malheureux ordinateurs dotés d'adresses de seconde zone ne pourraient jamais naviguer sur l'Internet : en effet, une communication aussi simple que l'accès à un serveur Web demande que les paquets comportent une adresse source et une adresse destination valides, ne serait-ce que pour que le serveur puisse renvoyer au client Web le contenu de la page qu'il a voulu consulter. D'ailleurs dans un réseau fermé sans connexion à l'Internet les possibilités de communication sont limitées au réseau local, et c'est pour de tels réseaux qu'avaient été créées à l'origine les classes d'adresses non routables, que NAT a ensuite astucieusement détournées de leur destination, si j'ose dire.

Sur un réseau connecté à l'Internet qui ne contient que des postes de travail dotés d'adresses non routables, il y a au moins un nœud qui possède une adresse routable, c'est le routeur d'entrée du réseau, puisque justement il est connecté. Alors il y a au moins un moyen de faire communiquer un poste du réseau local avec l'extérieur : il faut pour cela que le routeur soit doté de la capacité de traduction d'adresses ; ainsi il pourra jouer vis-à-vis des nœuds du réseau local le même rôle que le standard de l'hôtel vis-à-vis des postes téléphoniques des chambres, en « passant les communications ». Le principe de NAT est de remplacer une adresse interne non routable par une adresse routable.

Réalisations

La façon la plus simple pour réaliser la traduction d'adresses est la méthode *statique* : à chaque adresse interne non routable on fait correspondre, bijectivement, une adresse routable qui la remplace. Le routeur contient la table de correspondance et fait la traduction, sans autre forme de procès.

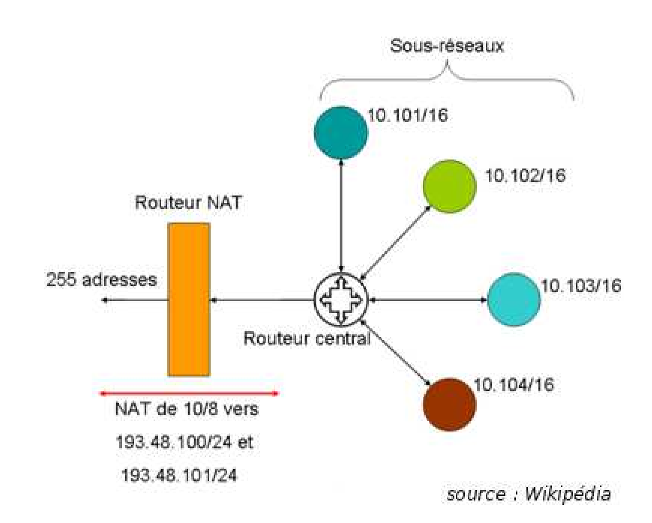


Figure 6.17
Réseau avec NAT : les adresses des hôtes sont des adresses réutilisables. Le routeur d'entrée fait la traduction d'adresses. On notera que la modification du plan d'adressage alloue désormais un réseau /16 par sous-réseau, s'affranchissant de la limite des 254 adresses possibles avec un /24.

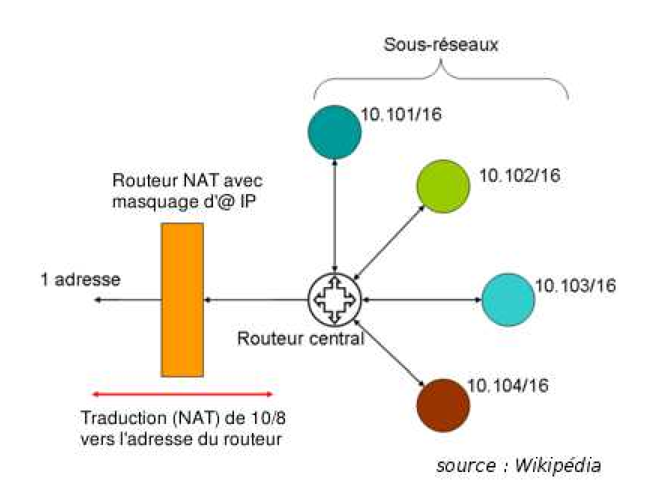
La traduction d'adresses statique est simple, mais dans l'univers de la fin des années 1990 la pénurie des adresses IP (la version 4 du protocole IP comporte des adresses sur 32 chiffres binaires, ce qui autorise un maximum de 4 294 967 295 adresses uniques, en fait un peu moins compte tenu des contraintes sur la structure de ces adresses, c'est-à-dire nettement moins que d'êtres humains à la surface de la Terre) a conduit vers d'autres réalisations, notamment la traduction d'adresses dite *dynamique*, et plus particulièrement vers une de ces méthodes dynamiques, dite *IP masquerading* (masquage d'adresse IP), aujourd'hui prédominante et que nous allons décrire brièvement (pour plus de détails et de références, cf. Wikipédia[8]). Avec NAT et le masquage d'adresse IP, seul le routeur possède une adresse routable, toutes les communications des nœuds internes sont vues de l'extérieur comme issues de cette adresse ou destinées à elle, et le tri est fait par le routeur au

[8] `http://fr.wikipedia.org/wiki/NAT`

moyen d'une manipulation des numéros de port, de façon tout à fait analogue au travail du standardiste de l'hôtel que nous évoquions ci-dessus.

Cette façon de recevoir à une adresse unique les paquets destinés en réalité à plusieurs machines, et de regrouper comme s'ils provenaient d'une adresse unique les paquets émis par ces diverses machines avant de les lancer dans l'Internet, s'appelle le *multiplexage*. À l'inverse, lorsque le routeur trie les les paquets reçus de l'Internet à son adresse pour les distribuer à leurs destinataires réels, il effectue un *démultiplexage*. Plus généralement, on nomme multiplexage le fait d'acheminer plusieurs communications sur un même support physique ou logique.

Figure 6.18
Réseau avec NAT et masquage d'adresse IP : seule l'adresse de l'interface externe du routeur est utilisée ; le multiplexage/démultiplexage des adresses IP internes se fait grâce aux numéros de ports (modifiés par le routeur).



On a vu (encadré page 121) qu'une connexion TCP était identifiée par le quadruplet {*adresse IP de destination, numéro de port de destination, adresse IP d'origine, numéro de port d'origine*}. En général, dans le paquet qui initie la connexion, le numéro de port de destination obéit à une convention (par exemple 80 pour l'accès à un serveur Web), et le numéro de port d'origine est quelconque, supérieur à 1 024, et choisi de façon à former un couple unique avec l'adresse d'origine. Lorsque le routeur recevra un tel paquet, où l'adresse d'origine sera une adresse NAT non routable, il remplacera cette adresse par sa propre adresse, éventuellement il remplacera le numéro de port d'origine par un autre, s'il a déjà utilisé ce couple {*adresse, numéro de port*} pour une autre traduction, et il conservera dans une

table la correspondance entre ce couple {*adresse, port*} envoyé sur l'Internet et celui du poste émetteur, ce qui permettra, au prix donc d'une traduction, d'acheminer les paquets dans les deux sens.

Sur l'unicité des *sockets*

Il est possible, lors de l'initiation d'une connexion réseau, de déterminer un tel couple {*adresse,port*}, nommé *socket*. Celui-ci est doté de la propriété d'unicité, car ce n'est pas le logiciel client qui établit la connexion, mais le *noyau du système d'exploitation*, du moins dans les systèmes sérieux (certains systèmes rudimentaires d'un passé récent ont pu implanter tout ou partie de la pile réseau dans l'espace utilisateur, ce qui ouvrait la voie aux pannes et aux piratages).

Une solution, quelques problèmes

À première vue, NAT est une *solution* de sécurité : avec un tel procédé et le masquage d'adresses IP, les adresses des nœuds du réseau interne, qui sont en général les postes de travail des utilisateurs, ne sont pas visibles de l'extérieur, ces nœuds sont donc hors d'atteinte de connexions éventuellement établies par des malfaisants, et de fait il n'y a en général aucune raison valable pour qu'une connexion soit établie depuis l'extérieur vers un poste de travail individuel ; si tel devait être le cas, cela devrait être fait selon une méthode de traduction explicite, par exemple pour permettre la prise de contrôle à distance dans un contexte d'assistance technique ou d'administration du système (mise à jour d'antivirus, etc.).

Cette protection du réseau privé par NAT est réelle et elle ne doit pas être sous-estimée. Il convient cependant d'avoir conscience du fait que, avec la version 6 du protocole TCP/IP, NAT va probablement disparaître, au moins sous sa forme actuelle, et avec lui les politiques de sécurité qui reposeraient trop fortement sur ses caractéristiques contingentes.

De toute façon, NAT n'est pas une solution très orthodoxe du point de vue de l'architecture du réseau, et peut même apparaître comme un *problème* de sécurité :

1. NAT viole le principe d'indépendance des couches du protocole, en effet le routage dans le réseau (IP, couche 3) au travers d'un routeur NAT utilise et modifie des informations contenues dans les en-tête de la couche transport (TCP ou UDP, couche 4), en l'occurrence les numéros de port ;

2. NAT viole le principe de *connectivité de bout en bout,* fondamental dans IP, qui signifie que toute l'« intelligence » du réseau doit être concentrée dans les nœuds terminaux, et que le réseau lui-même doit assurer un transport neutre ; avec NAT le routeur qui assure la traduction excède son rôle canonique ;
3. le routeur NAT conserve une trace des échanges sur le réseau, ce qui revient à dire que tous les échanges sont effectués en *mode connecté*, or IP admet des protocoles non connectés, tel UDP, qui subit ainsi une transformation insidieuse en protocole connecté ;
4. les en-tête TCP et UDP comportent des numéros de port, mais aucune loi n'interdit d'utiliser au-dessus de la couche IP d'autres protocoles de transport, qui pourraient ne pas reposer sur le concept de port, et qui de ce fait ne pourraient pas franchir un routeur NAT.

Protocoles connectés et non connectés

Dire qu'UDP est un protocole *non connecté* signifie qu'il émet chacun de ses paquets sans conserver à son sujet d'information d'état : aussitôt émis, aussitôt oublié. À l'inverse, TCP est un *protocole connecté*, ce qui signifie qu'il tient un journal des paquets émis ou reçus, de façon notamment à vérifier qu'ils sont dans le bon ordre et tous bien reçus, ce dont UDP n'a cure.

NAT pose des problèmes aux protocoles qui transportent des adresses IP et des numéros de port dans la partie « données » de leurs paquets. De tels protocoles sont dits « sales », parce qu'ils ne respectent pas le modèle d'abstraction en couches, et qu'ils transportent de l'information de niveau protocolaire (adresses) sous forme de données quelconques. Le type même du protocole sale est H323, utilisé pour la téléphonie sur IP et la visioconférence. Pour franchir un routeur NAT, un tel protocole doit être implémenté de façon que le routeur puisse inspecter le contenu des paquets et traduire les adresses qui s'y trouvent, puis recalculer la somme de contrôle et la longueur du paquet.

NAT pose aussi des problèmes à IPSec, il est en fait rigoureusement incompatible avec le protocole AH d'IPSec car il modifie les adresses et les numéros de ports.

NAT modifie donc les paquets, ce qui, du moins en IPv4, oblige à recalculer la somme de contrôle qui y figure (IPv6 supprime cette contrainte).

Dans un réseau qui met en œuvre NAT, le masquage d'adresse IP et les adresses non routables de la RFC 1918 (cf. la figure 6.18), ce qui est très répandu, notamment avec les petits routeurs ADSL que chacun installe maintenant à son domicile, les adresses sont généralement affectées de façon dynamique par un protocole conçu à cet effet, DHCP *(Dynamic Host Configuration Protocol)*. Ce protocole n'est pas exempt de critiques du point de vue de la sécurité, notamment parce qu'il émet des diffusions générales à la cantonade sans que ce soit toujours nécessaire, et aussi parce qu'il n'est pas protégé contre les usurpations d'identité : je monte un serveur DHCP pirate, j'alloue aux clients naïfs des adresses que je contrôle, je fais croire au service de noms local que les communications à destination de ces adresses doivent m'être envoyées, et ainsi j'intercepte des communications qui ne me sont pas destinées.

Promiscuité sur un réseau local

Lorsqu'il est question de sécurité du réseau, le plus souvent on pense à la protection contre les attaques en provenance de l'Internet. Or négliger les attaques en provenance de l'intérieur par le réseau local *(Local Area Network, LAN)* serait s'exposer à des menaces qui deviennent de jour en jour plus réelles avec le développement du nomadisme et des réseaux sans fil, et qui d'ailleurs existaient de tout temps. De ce point de vue nous pouvons dire que les réseaux sans fil ne produisent aucune menace qui n'ait déjà existé, ils ne font que susciter la prise de conscience des risques qui en résultent, et bien sûr en accroître l'intensité.

Nous allons le voir, il règne sur un réseau local une véritable *promiscuité*, au sens où un utilisateur mal intentionné dispose de certains moyens d'accès direct aux communications qui ne lui sont pas destinées.

Rappel sur les réseaux locaux

On nomme habituellement réseau local une infrastructure de couche 2 (liaison de données) qui dessert un bâtiment ou un campus. Une telle infrastructure comporte en général, outre le câblage, des répéteurs, des commutateurs et des bornes d'accès pour réseaux sans fil, mais pas de routeur, hormis celui qui relie le réseau local à l'Internet. C'est le schéma classique de l'équipement d'un site d'entreprise.

Nous n'évoquerons ici que les réseaux locaux définis par la norme IEEE 802.3, plus communément nommés *Ethernet*, puisque les autres types de réseaux définis par ce groupe de normes ne sont plus guère utilisés. Disons tout de suite que les réseaux sans fil 802.11 (dits *Wi-Fi*) reposent par bien des points sur les mêmes principes techniques que 802.3, notamment pour leurs caractéristiques significatives du point de vue de la sécurité.

La norme 802.3 décrit des réseaux où toutes les stations partagent un support physique unique ; à l'origine il s'agissait d'un câble coaxial sur lequel toutes les stations étaient branchées en émission comme en écoute (câblage dit 10Base5), puis apparurent des répéteurs *(hubs)* auxquels les stations étaient reliées par des paires téléphoniques torsadées (câblage dit 10BaseT), mais toutes les données circulant sur le réseau atteignaient toutes les stations. Aujourd'hui la plupart des réseaux utilisent un câblage 100BaseT ou 1 000BaseT en étoile autour de commutateurs *(switches)*, qui sont des répéteurs « intelligents » capables d'« apprendre » sur quelle branche de l'étoile se trouve telle station, ce qui leur permet d'établir des liaisons point à point et améliore ainsi considérablement la sécurité des communications. On peut dire que les réseaux 802.11 font revivre la première époque d'Ethernet 802.3, où toutes les stations accédaient au même support physique et pouvaient, de ce fait, recevoir *toutes* les données échangées sur ce support.

Une autre conséquence du partage du support physique par toutes les stations, c'est que deux stations peuvent essayer d'émettre simultanément, avec pour résultat ce que l'on appelle une *collision*, qui provoquera le brouillage des communications. Pour résoudre ce problème, les stations d'un réseau 802.3 mettent en œuvre le protocole dit *Carrier Sense Multiple Access with Collision Detection (CSMA-CD)*, ou accès multiple par écoute de la porteuse, avec détection de collision. De leur côté, les réseaux 802.11 ont recours au protocole *Carrier Sense Multiple Access with Collision Avoidance (CSMA-CA)*, analogue à CSMA-CD, mais avec évitement des collisions, parce que sur un réseau sans fil les collisions ne peuvent pas toujours être détectées, du fait que chaque station ne « voit » pas forcément toutes les autres. La description de ces protocoles excède le cadre du présent exposé ; on pourra plus se reporter à un exposé général [17], à une présentation technique détaillée des réseaux 802.11 [11], ou à un ouvrage de référence complet sur les réseaux informatiques [108].

VOCABULAIRE La porteuse

Les systèmes de transmission électro-magnétiques, avec ou sans fil, procèdent souvent par l'émission d'une onde sur une fréquence constante, le signal étant réalisé par une modification de cette fréquence, ou sa *modulation*. L'onde de fréquence constante est appelée la *porteuse* (*carrier* en anglais).

Réseaux locaux virtuels (VLAN)

Les réseaux locaux virtuels *(Virtual LAN, VLAN)* sont apparus en 1995, avec les commutateurs 802.3. Il s'agit donc d'un dispositif de couche 2 (liaison de données), en pratique Ethernet. L'idée est la suivante : il peut être tentant, notamment pour des raisons de sécurité, de regrouper les stations d'un groupe de personnes qui travaillent dans la même équipe sur un réseau local qui leur sera réservé, séparé des réseaux des autres équipes. Mais si les membres des différentes équipes sont dispersés dans différents bâtiments et mélangés avec les autres groupes, adapter le câblage physique à l'organisation peut se révéler coûteux et malcommode, d'autant plus que la répartition géographique des membres de chaque équipe peut changer. On a donc recherché des moyens de créer, sur une infrastructure parfois complexe, des réseaux locaux virtuels, qui isoleraient logiquement les communications propres à un groupe de stations, lequel partagerait tout ou partie d'un même support physique avec d'autres groupes. En somme il s'agit de faire au niveau de la couche 2 (liaison de données) ce que les VPN (voir page 114) font au niveau de la couche 3 (réseau).

Après quelques errements, les VLAN ont été normalisés en 1998 par la norme 802.1Q, qui a nécessité une modification du format de la trame Ethernet afin de lui ajouter 4 octets, dont 12 bits constituent une étiquette *(tag)* destinée à identifier les trames qui appartiennent à tel ou tel réseau local virtuel. Les commutateurs modernes sont programmés pour tenir compte de ces étiquettes, et pour n'acheminer les trames que vers des destinations qui appartiennent au VLAN désigné par leur étiquette.

Ce sont les commutateurs qui jouent le rôle principal dans la gestion des VLAN : le premier commutateur que rencontre une trame lui affecte une étiquette, qui déterminera son VLAN, et, partant, son destin.

Un lien physique partagé par plusieurs VLAN est nommé *trunk* dans le jargon des VLAN, ou *channel* dans la terminologie du constructeur *Cisco*. Cette divergence terminologique illustre un inconvénient des VLAN : il s'agit d'une technologie implantée dans les méthodes de configuration des commutateurs propres à chaque constructeur, et de ce fait difficile à exposer en concepts clairs.

Il est de bonne politique que le routeur de sortie du réseau vers l'Internet appartienne à tous les VLAN, ou du moins à tous ceux dont les stations doivent pouvoir atteindre l'Internet. Exclure ce routeur d'un VLAN est un bon moyen d'interdire aux utilisateurs de ce VLAN de naviguer sur l'Internet.

Les VLAN peuvent être utiles en termes de sécurité, par exemple en limitant la promiscuité sur un réseau local. Une application assez répandue et commode de ce procédé consiste, sur un campus ou au sein d'une entreprise, à créer pour accueillir les ordinateurs portables des visiteurs extérieurs un VLAN où ils seront confinés, ce qui évitera qu'ils puissent accéder aux serveurs internes, ou qu'ils répandent dans l'entreprise les virus dont ils pourraient être infectés, tout en ayant la possibilité d'accéder à l'Internet ou à toute autre ressource qui leur aura été autorisée.

Sécurité du réseau de campus : VLAN ou VPN ?

Nous venons de voir que les VLAN permettaient d'améliorer la sécurité d'un réseau local en cloisonnant le trafic réseau par la réservation à chaque équipe ou entité fonctionnelle d'un réseau privé virtuel, et en limitant ainsi la promiscuité des données.

Une autre façon de segmenter le réseau est de recourir à des routeurs. Nous avons vu ci-dessus (page 114) que les réseaux privés virtuels (VPN) permettaient d'établir des tunnels chiffrés entre deux stations quelconques sur l'Internet.

Comment choisir entre ces deux types de solution ?

Les VLAN, par définition, ne peuvent être déployés qu'au sein d'un même réseau local. Dans ce rôle ils sont très commodes : une fois les commutateurs configurés, tout est automatique. Les commutateurs sont plus faciles à configurer que les routeurs, ils sont aussi moins chers et plus rapides. Les réseaux commutés demandent moins de compétences humaines et moins d'investissements matériels que les réseaux routés. Leur inconvénient principal, malgré la promulgation de la norme 802.1Q, est de reposer le plus souvent sur des recettes de configuration propres

à chaque constructeur, qui violent plus ou moins ouvertement le principe de l'indépendance protocolaire : les VLAN mélangent des fonctions qui relèvent de la couche 2 avec des fonctions de couche 3. Cette confusion n'a pas que des inconvénients théoriques, elle peut conduire à l'édification d'un réseau à la topologie confuse, dont l'évolution ultérieure et la maintenance seront difficiles.

Le routage est une technique qui repose sur des bases théoriques et conceptuelles solides et acceptées par tous. En fait, il est la pierre angulaire de l'Internet. Les protocoles privés créés naguère par certains constructeurs cèdent de plus en plus souvent la place aux protocoles normalisés et documentés, tel OSPF *(Open Shortest Path First)*[9]. Un réseau privé virtuel peut s'étendre, virtuellement donc, à l'ensemble de la planète, mais il est aussi tout à fait possible de construire pour un coût marginal un minuscule VPN entre mon ordinateur au bureau, celui de mon domicile et mon ordinateur portable connecté à un point d'accès sans fil.

En pratique

Nous pensons qu'il est très intéressant, sur un campus, de créer un VLAN pour accueillir les ordinateurs portables des visiteurs auxquels on ne veut pas accorder de droits, mais qui doivent quand même travailler et accéder à l'Internet, ne serait-ce que pour communiquer avec leurs bases. Pour tout autre usage, il faut bien se demander si le VLAN ne serait pas une solution paresseuse à un problème pour lequel le routage serait plus satisfaisant.

Réseaux sans fil et sécurité

Les réseaux sans fil connaissent un engouement important, et les problèmes de sécurité qu'ils soulèvent ont fait l'objet d'une note de synthèse du Centre d'expertise gouvernemental de réponse et de traitement des attaques informatiques (CERTA)[10]. Le cadre réglementaire de leur mise en œuvre est résumé par un texte de l'Autorité de régulation des communications électroniques et des postes (ARCEP)[11]. Mais commençons par délimiter ce dont il s'agit.

[9] *Open Shortest Path First* (OSPF) est un protocole de routage basé sur un algorithme de recherche de parcours dans un graphe dû à Dijkstra.

[10] `http://www.certa.ssi.gouv.fr/site/CERTA-2002-REC-002/`

[11] `http://www.art-telecom.fr/dossiers/rlan/schema-rlan.htm`

Types de réseaux sans fil

Les réseaux sans fil appartiennent à plusieurs catégories, régies par des normes spécifiques :

- les réseaux dits *Wireless Local Area Network (WLAN)* ou *Wi-Fi* obéissent aux normes de la famille IEEE 802.11, dont la première édition date de 1997 ; ils sont destinés à faire communiquer des équipements séparés par une distance de l'ordre de quelques dizaines de mètres, par exemple dans un immeuble ; les dispositifs d'émission et de réception de ces appareils ont une puissance maximale de 100 mW (à comparer avec celle d'un téléphone portable GSM, qui est de 1 W) ;
- les réseaux dits *Wireless Personal Area Network (WPAN)* ou *Bluetooth* obéissent à la norme IEEE 802.15.1 ; ils permettent des communications entre des appareils distants de quelques mètres, par exemple un téléphone et son oreillette sans fil ; les promoteurs de cette norme l'ont déjà déployée pour les assistants personnels (PDA) et ils envisagent des débouchés sur le marché du jouet et des consoles de jeu ; la puissance des émetteurs est plus faible que pour les appareils 802.11, en général 1 mW (il existe bien une option de la norme qui permet une puissance de 100 mW, mais elle n'est pratiquement pas utilisée), et de ce fait la consommation électrique est moindre ; la norme IEEE 802.15.3 (Bluetooth2) est une évolution de la norme Bluetooth avec des débits plus rapides et des mécanismes de sécurité améliorés par rapport à 802.15.1 ;
- les réseaux dits *Wireless Metropolitan Area Network (WMAN)* obéissent à la norme 802.16, plus connue sous le nom de WiMax, ou de *Boucle locale radio (BLR)* ; ils sont capables de relier des équipements distants de quelques kilomètres, par exemple pour se substituer aux liaisons ADSL dans les zones rurales à faible densité ;
- les réseaux dits *Wireless Wide Area Network (WWAN)* utilisent les systèmes de téléphonie sans fil tels que *GSM (Global System for Mobile Communication), GPRS (General Packet Radio Service)* ou *UMTS (Universal Mobile Telecommunication System)* comme couche de liaison de données pour constituer une infrastructure d'accès à l'Internet.

Nous nous intéresserons surtout ici aux réseaux 802.11, dont on trouvera une description technique détaillée sur le site des Journées réseau de l'enseignement supérieur (JRES) 2005 [12].

Vulnérabilités des réseaux sans fil 802.11

Tout d'abord, les réseaux 802.11 sont affectés de toutes les vulnérabilités qui concernent les réseaux 802.3. En effet, capter les signaux hertziens d'un réseau 802.11 procure les mêmes moyens d'observation et d'action que l'accès en mode *promiscuous* au support physique d'un réseau local câblé. Simplement, procéder à ce type d'intrusion est plus facile et plus discret. Le remplacement des répéteurs par des commutateurs avait diminué la vulnérabilité des réseaux câblés en établissant des liaisons point à point : le support hertzien ramène le candidat à l'intrusion au temps des répéteurs, où le mode *promiscuous* donnait accès à toutes les trames de toutes les stations.

Vulnérabilités 802.11 spécifiques

Le Centre d'expertise gouvernemental de réponse et de traitement des attaques informatiques (CERTA) énumère trois types de vulnérabilités propres aux réseaux sans fil :

1. la diffusion de l'information facilitant l'interception passive à distance ;
2. la sensibilité au brouillage diminuant la disponibilité du réseau ;
3. les configurations non sécurisées par défaut des nouveaux équipements, facilitant les attaques.

Les articles `http://tinyurl.com/hkb76` et `http://tinyurl.com/nwyu7` signalent en outre la possibilité de prendre à distance le contrôle d'un ordinateur portable en exploitant les failles de leurs pilotes Wi-Fi.

Protection des points d'accès

Pour la sécurité des points d'accès, les mesures suivantes sont préconisées :

1. changer les mots de passe par défaut (notamment administrateur) par des mots de passe plus forts ;
2. désactiver les services disponibles non utilisés (SNMP, Telnet...) ;

[12] `http://2005.jres.org/tutoriel/Reseaux_sans_fil.livre.pdf`

3. régler la puissance d'émission du point d'accès au minimum nécessaire ;
4. changer l'identifiant de réseau (SSID) par défaut ;
5. mettre à jour le *firmware* de son point d'accès dès que le constructeur propose une mise à jour.

Protection des communications par chiffrement

La facilité d'écoute des communications hertziennes suggère fortement de chiffrer le trafic 802.11. La norme propose une méthode de chiffrement, *Wired Equivalent Privacy (WEP)*, qui est malheureusement « inapte à offrir un niveau de sécurité suffisant pour la plupart des utilisateurs. En effet, il est possible en écoutant une quantité suffisante de trafic (cela peut prendre plusieurs heures selon l'activité du réseau), de casser une clé WEP en quelques secondes. Une documentation abondante sur le sujet est disponible sur l'Internet. Plusieurs outils d'attaque publics permettent de faire cela facilement, sans matériel spécialisé, dans un temps raisonnable », nous dit le document du CERTA[13].

En juin 2004 l'IEEE a ratifié la norme 802.11i, qui comporte de nouveaux protocoles de sécurité, plus robustes :

1. *Wi-Fi Protected Access (WPA)*, qui propose deux modes de fonctionnement : *WPA-PSK Mode* repose sur un secret partagé, cependant que *WPA Enterprise Mode* utilise le protocole d'authentification RADIUS ;
2. dans les deux cas, le chiffrement est effectué selon l'algorithme *Advanced Encryption Standard – FIPS-197 (AES)*, déjà mentionné à la page 72 comme un protocole symétrique robuste.

Le fonctionnement du protocole RADIUS

Le protocole RADIUS *(Remote Authentication Dial In User Service)* décrit un principe d'authentification très général : un individu souhaite accéder à un service en réseau pour lequel il lui faut s'authentifier ; pour ce faire il va envoyer ses données d'authentification (couple identifiant-mot de passe, ou certificat électronique, par exemple) à un serveur RADIUS, qui lui-même établira une transaction avec le véritable serveur d'authentification (annuaire électronique, ou système de mot de passe d'un serveur Unix...). Le protocole RADIUS permet ainsi d'utiliser des systèmes d'authentification préexistants pour de nouvelles applications en réseau, sans avoir à modifier ni le serveur ni l'application.

[13] http://www.certa.ssi.gouv.fr/site/CERTA-2002-REC-002/

Authentification des accès au réseau

La norme 802.11 définit une méthode d'authentification par défi-réponse, dite *Shared Key Authentication* :

1. le mobile qui veut accéder au réseau envoie au système de contrôle d'accès du point d'accès (qui peut être un routeur ou une borne Wi-Fi) une demande d'authentification ;
2. le point d'accès envoie un texte aléatoire au mobile ;
3. le mobile chiffre ce texte avec sa clé WEP et envoie le texte chiffré au point d'accès ;
4. le point d'accès déchiffre le message avec la clé WEP censée correspondre à celle du mobile et le compare au texte original : s'ils coïncident, c'est que le mobile et le point d'accès partagent la même clé WEP, et l'accès est accordé.

Il est également possible de restreindre l'accès aux adresses MAC autorisées ; l'adresse MAC (comme *Medium Access Control*) d'un équipement connecté à un réseau local, avec ou sans fil, est son adresse de couche 2 ; en général elle est enregistrée « en dur » dans la carte réseau, ce pourquoi on l'appelle souvent adresse physique, ce qui ne veut pas dire pour autant qu'elle soit infalsifiable, mais dans des conditions normales elle n'est jamais modifiée. Ces deux méthodes sont considérées comme insuffisantes, parce qu'il existe des méthodes de contournement faciles à mettre en œuvre. Aussi est-il conseillé d'employer une méthode plus robuste, et susceptible d'ailleurs d'être utilisée également pour un réseau câblé 802.3 : celle définie par la norme IEEE 802.1x.

La norme IEEE 802.1X définit une méthode générale d'authentification des accès à un point d'entrée du réseau, qu'il s'agisse d'une prise d'un commutateur filaire ou d'un point d'accès sans fil. Le protocole d'authentification proprement dit est laissé au choix de l'administrateur du système parmi les variantes d'EAP *(Extensible Authentication Protocol)* :

- EAP-TLS (*EAP - Transport Layer Security*, RFC 2716) ouvre entre le mobile et le point d'accès un tunnel sûr dont l'accès est contrôlé par des certificats électroniques délivrés par une Infrastructure de gestion de clés (IGC, PKI en anglais) ;
- EAP-MD5 repose sur le chiffrement par le protocole MD5, notoirement fragile ;

- EAP-TTLS *(EAP-Tunneled Transport Layer Security)*, assez similaire à EAP-TLS, si ce n'est qu'il n'utilise qu'un seul certificat du côté du serveur ;
- PEAP *(Protected Extensible Authentication Protocol)*, très semblable au précédent ;
- et deux ou trois autres variantes plus ou moins désuètes.

Le protocole d'authentification EAP est encapsulé au-dessus du protocole IEEE 802.11. L'équipement d'accès au réseau sans fil (point d'accès) relaie les trames entre le client et le serveur d'authentification (serveur RADIUS), sans connaître le protocole EAP utilisé. Dans le cas où le protocole d'authentification prend en charge la gestion des clés, celles-ci sont transmises à l'équipement d'accès puis au client dans le cadre du chiffrement. Le protocole d'authentification le mieux adapté semble être EAP-TLS *(EAP - Transport Layer Security)* créé par Microsoft et accepté sous la norme RFC 2716, ce qui, curieusement, introduit une touche de confusion protocolaire.

Une alternative à la sécurisation des réseaux Wi-Fi

La mise en oeuvre de solutions de sécurité Wi-Fi de type WPA reste complexe : choix de bornes Wi-Fi adéquates, serveur de sécurité, gestion des utilisateurs et éventuellement logiciel spécifique à installer sur les postes de travail.

Ajoutez à cela l'incertitude qui semble peser sur la sécurité effective de tels dispositifs. Une méthode assez simple qui est parfois employée pour déployer de tels accès en entreprise est de considérer l'accès Wi-Fi à l'identique d'un accès Internet. Le montage est alors le suivant :

- déploiement d'un réseau Wi-Fi peu protégé (un peu quand même pour limiter l'accès à des touristes) ;
- des filtres en sortie du réseau Wi-Fi n'autorisent le trafic que vers la passerelle VPN des utilisateurs nomades de l'entreprise, et cela afin d'éviter que les touristes ne puissent profiter de l'accès ;
- l'utilisateur ainsi connecté en Wi-Fi doit utiliser son logiciel VPN pour se connecter au réseau d'entreprise, technologie dans laquelle l'entreprise est en général plus confiante.

Il est bien sûr indispensable dans une telle situation que la passerelle VPN soit dimensionnée de façon adéquate, les volumes de données à traiter pouvant être importants.

L'accueil des visiteurs sur un réseau sans fil

De nombreuses entreprises reçoivent des visiteurs. Pouvoir leur donner un accès à l'Internet sans pour autant autoriser l'accès au réseau d'entreprise fait partie de ces petits plus qui facilitent la vie à tout le monde.

La généralisation des réseaux Wi-Fi ne fait qu'augmenter la pression des visiteurs pour bénéficier d'un tel service. Les avantages pour l'entreprise sont réels : il n'est plus nécessaire de gérer un câblage et des prises spécifiques pour les visiteurs, une infrastructure d'accueil sans fil permettra un accès depuis toute la zone de couverture.

Le déploiement d'un accès Wi-Fi pour les visiteurs nécessite de faire attention à quelques points :

- la simplicité d'utilisation ; on évitera donc des technologies compliquées comme celles basées sur WPA *(Wifi Protected Access)* qui nécessitent une prise en charge logicielle adéquate sur chaque poste de travail ; on préférera un portail Web permettant au visiteur de s'identifier, d'accepter les conditions d'utilisation et enfin d'utiliser le réseau Internet indispensable au travail de tous les jours ;
- l'information du visiteur sur ses droits, devoirs ainsi que sur les risques liés à l'utilisation d'un réseau sans fil est indispensable ;
- le respect de la réglementation en vigueur : l'opérateur d'un point d'accès à l'Internet pour ses visiteurs est considéré comme un fournisseur d'accès à l'Internet et doit donc enregistrer un certain nombre de données afin de permettre l'identification d'un utilisateur, par exemple dans le cadre d'une réquisition judiciaire.

Une architecture simple que l'on peut rencontrer (en le faisant soi-même ou bien en utilisant des produits du commerce) permet de mettre en œuvre un réseau sans fil utilisable aussi bien pour les besoins de l'entreprise que pour ceux des visiteurs. La contrainte essentielle est de disposer de points d'accès sans fil qui prennent en charge les réseaux virtuels multiples (VLAN) afin de séparer le trafic en plus d'une gestion de plusieurs SSID. On associera bien sûr un SSID à un VLAN pour chaque type d'utilisateur. Souvent, les produits d'entrée de gamme ne répondent pas à ces caractéristiques techniques : il convient de bien vérifier auprès de son fournisseur avant tout achat.

Le déploiement se fera ensuite selon les principes directeurs suivants :

- sur son réseau d'entreprise créer un VLAN pour les visiteurs – ce réseau virtuel n'aura pas accès aux ressources internes à l'entreprise, mais il devra disposer d'un accès à l'Internet (moyennant parfois des filtres de sécurité, l'entreprise reste maîtresse de la nature ouverte ou restreinte du service qu'elle propose) ;
- sur l'infrastructure Wi-Fi, transporter ce VLAN sous un identifiant SSID dédié à cet usage ;
- configurer les bornes Wi-Fi afin que les différents utilisateurs du réseau visiteurs ne puissent pas se voir (lorsqu'il y a une seule borne Wi-Fi c'est assez simple à faire, par exemple sur une borne *Aironet* de *Cisco* on utilisera la fonction *Public Secure Packet Forward*) ;

- mettre en oeuvre un *portail Wi-Fi* pour les visiteurs, qui assure l'identification et l'information de l'utilisateur avant de lui donner accès à l'Internet – l'enregistrement et l'attribution des codes d'accès pourront se faire lors de l'accueil du visiteur ou bien à l'initiative des salariés disposant d'un accès au portail d'enregistrement.

Le marché propose aujourd'hui des solutions permettant de mettre en œuvre de telles solutions d'accès, et bien sûr les utilisateurs expérimentés pourront souhaiter déployer eux-mêmes une solution à base d'outils libres ; un serveur Linux avec quelques logiciels adéquats permet de mettre en œuvre un tel service :

- un portail Web permet (depuis le réseau filaire) aux personnes autorisées de créer des accès pour les visiteurs et d'imprimer leur fiche d'accès ;
- lorsque le visiteur utilise le réseau sans fil, il ne peut accéder directement à l'Internet, et le serveur « portail visiteurs » redirige toutes les connexions sortantes (au moins les connexions Web) vers le portail d'identification qui demandera au visiteur son code d'accès et le mot de passe associé ;
- une fois l'identité du visiteur vérifiée, le portail visiteurs autorise l'accès à l'Internet en configurant dynamiquement le pare-feu (*Netfilter* avec un système Linux) pour autoriser ce poste de travail, et lui seul, à sortir sur l'Internet ;
- lorsque le visiteur quitte l'entreprise, ou bien en fin de journée, le code d'accès est automatiquement désactivé ;
- le logiciel du portail garde trace des heures d'utilisation et des adresses IP attribuées : cela permet de répondre, si besoin est, à une réquisition judiciaire pour identifier l'internaute.

L'administrateur du réseau ne doit en outre pas oublier que lorsqu'il est dans une zone relativement dense, des tiers ne manqueront pas d'essayer, avec ou sans intention malveillante, d'accéder à son réseau « visiteurs », car celui-ci sera présenté comme étant « ouvert » et donc accessible à tous. Cela peut générer une quantité non négligeable d'événements dans les journaux d'activité. On peut aussi envisager, si les systèmes en place le permettent, des mécanismes de défense qui refusent l'accès à une station de travail après un certain nombre de tentatives d'accès.

7

Identités, annuaires, habilitations

Les questions d'identité et d'identifiants ne sont bien sûr pas totalement étrangères à l'univers de la sécurité, qui comporte notamment la question de savoir qui a le droit de faire quoi ; or pour répondre à cette question il faut pouvoir désigner qui et quoi avec exactitude et précision. Nous évoquerons dans ce chapitre la question des identifiants, les annuaires électroniques et les infrastructures de gestion de clés.

Qu'est-ce que l'identité dans un monde numérique ?

Cette section consacrée aux identifiants numériques doit beaucoup à la communication de Sophie Le Pallec [72] consacrée à ce sujet au congrès JRES 2005.

Problématique de l'identification

La question des identifiants est au cœur de la problématique de traitement des données, pour la raison évidente qu'une donnée ne présente d'intérêt que si l'on est capable de la distinguer des autres informations, et de distinguer l'entité ou les entités auxquelles elle se rapporte des autres entités présentes dans l'univers étudié, ce qui définit le processus d'identification. Un identifiant est constitué d'un ou plusieurs attributs qui permettent de distinguer une entité d'autres entités, c'est-à-dire de connaître son identité, parce que comme le dit Sophie Le Pallec « les entités sont des choses qui existent et qui peuvent être distinguées les unes des autres ».

Cette position de la question des identifiants au centre de la problématique informatique lui confère *ipso facto* un rôle tout aussi essentiel pour la sécurité des systèmes d'information, puisqu'il devient alors crucial de pouvoir être sûr d'une identité, de pouvoir vérifier l'authenticité d'un identifiant qui sert à alléguer cette identité.

Sophie Le Pallec pense (et elle n'est pas la seule) que de plus en plus d'objets de la vie courante ou de l'activité économique seront dans un proche avenir dotés d'identifiants numériques, utilisables par des procédés informatiques et accessibles par l'Internet : « ... nous nous avançons vers une ère où les plus petits objets qui nous entourent seront porteurs d'information et capables d'échanger cette information avec leur environnement. Il apparaît également comme acquis que l'infrastructure globale en charge de véhiculer cette information sera celle de l'Internet... »

Trois types d'usage des identifiants

Sophie Le Pallec distingue trois types d'usage des identifiants :

1. L'identifiant d'*immatriculation* est associé physiquement à l'entité identifiée, il procure un accès visuel à l'identité, que ce soit directement (nom de rue sur une plaque, code postal sur une enveloppe, numéro ISBN sur un livre, numéro d'immatriculation sur la plaque minéralogique d'une voiture) ou indirectement (numéro de sécurité sociale dans la puce d'une carte Vitale, numéro IMEI *(International Mobile Equipment Identity)* d'un téléphone mobile, code-barre EAN*UCC *(European Article Numbering - Uni-*

form Code Council) du paquet d'un produit alimentaire dans un rayon de supermarché).

2. L'identifiant d'*indexation* sert à caractériser de façon unique un enregistrement dans une base de données, il est une clé d'accès à l'information contenue dans cet enregistrement.
3. L'identifiant de *connexion* permet de repérer sans ambiguïté le chemin d'accès à une entité connectée au réseau : il importe en effet que la localisation d'une telle identité soit identifiée de manière unique, afin que la transmission et le routage de l'information se fassent sans équivoque vers le bon récepteur. Sophie Le Pallec en distingue deux variétés :
   - les identifiants de *connexion directe*, tels les numéros de téléphone du plan de numérotation international E.164, ou les adresses IP dans l'Internet ; il existe pour un réseau donné un seul espace d'identifiants de connexion directe ; de tels identifiants ne peuvent pas être considérés comme permanents, parce que si l'entité qu'ils repèrent se déplace dans le réseau ou change d'opérateur de réseau, elle changera de numéro de téléphone ou d'adresse IP ;
   - les identifiants de *connexion indirecte* sont des identifiants intermédiaires qui vont désigner un identifiant direct par le truchement d'un mécanisme de résolution ; ils sont généralement utilisés pour pallier le défaut de permanence des identifiants directs.

Notons enfin qu'un identifiant peut procurer :

- soit un accès direct à l'objet physique identifié : le numéro de téléphone permet d'atteindre le poste de l'abonné, l'adresse IP l'ordinateur auquel elle est attribuée ;
- soit l'accès à une *information* sur l'objet identifié : le numéro INSEE, dit improprement numéro de sécurité sociale, donne accès à certaines données relatives à la personne identifiée ;
- il existe enfin des usages hybrides de certains identifiants : le numéro ISBN (pour *International Standard Book Number* cf. `http://www.isbninternational.org/`) d'un livre permet, dans une bibliothèque dont le système d'information donne la correspondance entre cote et ISBN, de retrouver sa notice dans un catalogue, mais aussi de le retrouver dans son rayon, ou de le commander chez un libraire.

Vers un système universel d'identifiants

Des systèmes d'identifiants utilisés de façon générale (par opposition à des usages locaux, au sein d'une entreprise ou d'une région), ce qui suppose qu'un organisme d'enregistrement garantisse leur unicité, peuvent constituer des espaces d'identifiants à vocation universelle ; nous avons déjà cité le plan de numérotation téléphonique international E.164 de l'Union Internationale des Télécommunications (UIT), l'adressage IP dans l'Internet, ou les identifiants à usage logistique, comme le système commun créé par la fusion du système européen d'EAN International *(European Article Numbering)* et du système américain UCC *(Uniform Code Council)* pour former un espace unique d'identification des objets matériels tout au long de la chaîne d'approvisionnement, administré par le consortium GS1 (`http://www.gs1.org/`). Les activités de GS1, qui regroupe un million d'entreprises dans une centaine de pays, se manifestent jusque sur les rayons des supermarchés par le code-barre qui orne votre paquet de café ou votre bouteille d'eau minérale (le Château-Margaux et la Romanée-Conti y échappent encore). Citons également le DOI (pour *Digital Object Identifier*, cf. `http://www.doi.org`), issu du monde de la documentation.

La question se pose de la convergence de ces espaces d'identifiants : l'imbrication croissante des activités économiques et des échanges de données à l'échelle mondiale plaide pour l'unification. Dans une telle perspective le plan d'adressage IP de l'Internet semble bien placé pour l'emporter : comme E.164 de l'UIT il est incarné dans une infrastructure technique qui procure un accès direct à chaque entité identifiée, mais son architecture est nettement plus ouverte (chacun peut s'y agréger plus facilement), et surtout il dispose d'un double système identifiants de connexion directe (adresses IP) – identifiants de connexion indirecte (URI, pour *Uniform Resource Identifier, cf.* `http://fr.wikipedia.org/wiki/URI`). Les identifiants de connexion indirecte, ici les URI tels que `http://www.ietf.org`, constituent l'espace de nommage de l'Internet (par opposition à l'espace d'adressage), dans le fonctionnement duquel ils jouent plusieurs rôles :

- comme indiqué ci-dessus, ils pallient la non-permanence des adresses IP (à votre domicile, le plus souvent, votre fournisseur d'accès à l'Internet vous attribue une adresse IP différente à chaque nouvelle connexion et avec IPv6 cette variabilité des adresses IP se généralisera) ;
- le système de résolution des identifiants de connexion indirecte (URI) en identifiants de connexion directe (adresses IP), le DNS *(Domain Name Sys-*

tem), bénéficie d'un déploiement universel et d'une avance technique certaine sur ses concurrents éventuels, parmi lesquels un des mieux placés, la norme LDAP (pour *Lightweight Directory Access Protocol*) d'annuaire électronique, appartient en fait au même univers technique ;

- enfin, ils fournissent une identification facile à mémoriser, ce qui est important, notamment d'un point de vue commercial.

Des mécanismes d'abstraction du cerveau

Ici je ne résiste pas à la tentation de citer plus longuement Sophie Le Pallec : « Une abstraction opérée naturellement par le cerveau permet de passer, sans prise de conscience forte, de la fonction de l'identifiant d'immatriculation à celle d'identifiant d'indexation ou d'identifiant de connexion, lorsqu'un même identifiant, ou des identifiants sémantiquement proches, assurent ces trois fonctions. Ce mécanisme d'abstraction, qui est une des bases du langage, autorise la dualité fonctionnelle des identifiants [...], ainsi nous trouvons tout à fait normal que l'identifiant « soleil », appliqué à nommer l'astre en question, puisse être associé simultanément à l'information sur l'astre, qu'elle soit accessible via une base de donnée ou un moteur de recherche en tapant comme mot-clé « soleil ». De même, nous trouvons naturel qu'un site Web avec l'adresse `www.soleil.com` fasse dans son contenu référence à l'astre solaire. Alfred Korzybski, spécialiste de la sémantique, associe au mot « identification » l'expression « confusion des ordres d'abstractions ». Il s'agit pour lui de nommer une perturbation sémantique (« identification ») que l'on retrouve à la base des troubles mentaux et de décrire un processus sous-jacent, systématique, à cette perturbation (« confusion des ordres d'abstractions »). Il est intéressant de conserver à l'esprit que cette confusion peut être utilisée à dessein de manière tout à fait saine, mais pas forcément toujours très consciente, dans les processus d'identification complexes que nous pouvons mettre en œuvre. »

La politique des identifiants

Si l'adressage IP devient hégémonique, se posera la question du contrôle politique de son administration. Historiquement, celle-ci était dévolue à l'IANA *(Internet Assigned Numbers Authority)*, qui centralise et contrôle les conventions relatives à l'identification des objets du réseau, et notamment veille à l'unicité des adresses, mais depuis 1998 c'est l'*Internet Corporation for Assigned Names and Numbers (ICANN)* qui supervise l'attribution des noms de domaines et des adresses.

L'ICANN est une organisation internationale sans but lucratif dont le rôle est d'allouer l'espace des adresses du protocole Internet (IP), d'attribuer les identificateurs de protocole, de gérer le système de noms de domaine de premier niveau pour les codes génériques (gTLD, pour *generic Top Level Domain*, tels `.com`

ou `.edu`) et les codes nationaux (ccTLD, pour *country code Top Level Domain*, tels `.fr` ou `.be`), et d'assurer l'organisation et le fonctionnement du système de serveurs racines ; pour chacune de ces transactions, l'ICANN prélève une redevance, ce qui assure son financement. Dans la perspective d'une généralisation de l'adressage IP, on mesure que cette institution, de fait largement contrôlée par le gouvernement des États-Unis, disposera d'un pouvoir d'autant plus exorbitant que l'on ne voit pas très bien, dans l'organisation de l'Internet telle qu'elle existe aujourd'hui, ce qui pourrait le contrebalancer.

Distinguer noms et identifiants dans le DNS ?

Michael J. O'Donnell, du département d'informatique de l'université de Chicago [83], défend, non sans arguments, le point de vue suivant : dès lors que les noms de domaines ont une signification pour les êtres humains qui les lisent, les écrivent et s'en souviennent, ils sont inéluctablement susceptibles de devenir des enjeux politiques ou commerciaux.

Si la lutte pour un de ces enjeux que sont les noms de domaines aboutit à ce que le titulaire d'un nom en soit dépossédé, avec les règles actuelles d'administration du DNS ses visiteurs perdent tout moyen d'atteindre son site ou son adresse électronique. Pour éviter cela, O'Donnell propose la création d'une couche intermédiaire entre les noms et les adresses IP : les *handles (poignées)*, qui ne seraient en fait rien d'autre que des identifiants sans signification en langage humain, par exemple des séquences de chiffres. Ces identifiants seraient dotés des propriétés suivantes : gratuité, unicité, permanence, pérennité. Ces propriétés distinguent les identifiants proposés par O'Donnell des adresses IP, qui changent dès que le site ou le service se déplace géographiquement, change de fournisseur d'accès, et avec IPv6 les adresses seront encore plus volatiles.

Pour faire fonctionner ce système de poignées, il suffirait de les enregistrer dans un domaine particulier, réservé à cet effet, de l'actuel DNS : ainsi tous les logiciels et toutes les infrastructures nécessaires sont déjà en place, ce qui limite de façon drastique les investissements nécessaires à la mise en œuvre du projet. Et, comme les identifiants n'ont aucun contenu sémantique, ils ne devraient donner prise à aucune lutte pour leur possession.

Pretty Good Privacy (PGP) et signature

Le système PGP défraya la chronique judiciaire en 1993 lorsque son auteur Philip Zimmerman fut soumis à une enquête approfondie du FBI pour exportation illégale d'armes de guerre, en l'occurrence pour avoir placé son logiciel en accès libre sur l'Internet. Les autorités policières américaines (et françaises) ont tendance à penser que le chiffrement robuste est un obstacle à leurs investigations parce qu'il leur interdirait de déchiffrer les messages échangés par des criminels ou des ennemis. Aujourd'hui tous les dispositifs cryptographiques les plus puissants sont accessibles facilement par l'Internet et ainsi disponibles sans obstacles pour lesdits criminels et espions. Une législation restrictive n'entraverait par conséquent que les honnêtes citoyens soucieux de respecter la loi parce que c'est la loi, pas parce qu'il est difficile de faire autrement. Une telle législation n'aurait donc pour effet que de mettre les honnêtes gens à la merci des criminels, ce qui ne semble pas l'effet recherché, en principe du moins.

Sachant que de telles législations sont en déclin, même en France, pays qui a fermement tenu l'arrière-garde jusqu'en 1998, voyons le contenu de PGP. En fait, PGP n'apporte aucune révolution, il est plutôt un assemblage ingénieux et pratique des techniques évoquées au chapitre 4 page 71.

Pour pallier la lenteur des calculs d'algorithmes à la RSA, Zimmerman eut l'idée de recourir au bon vieux chiffrement à clé partagée ; comme le point faible de ce dernier est l'envoi de la clé, on utilisera RSA pour communiquer une clé de session pour un algorithme à clés symétriques, clé qui servira à chiffrer la suite des communications avec cet algorithme classique. En l'occurrence Zimmerman choisira IDEA, un cousin de DES à clés de 128 bits, créé à Zurich par James L. Massey et Xuejia Lai, et réputé très robuste. Notons que les systèmes de communication chiffrés tels que SSL *(Secure Socket Layer)* utilisés pour les transactions par le Web, la relève de courrier électronique et la connexion conversationnelle à distance par SSH *(Secure Shell)* fonctionnent de cette façon.

Cette utilisation combinée des méthodes de chiffrement symétrique (DES en l'occurrence) et asymétrique sera la vraie révolution pratique, qui suscitera la colère de la NSA et de ses homologues dans d'autres pays dont la France. Avant que cette possibilité n'existe, les utilisateurs de cryptosystèmes se mettaient laborieusement d'accord sur une clé, puis ils l'utilisaient pendant longtemps. La NSA disposait sans doute des moyens de casser le chiffrement DES, ce qui lui ouvrait des mois

de lecture paisible de messages réputés secrets. Avec la combinaison de DES et RSA, les utilisateurs changent de clé à chaque échange de messages, ce qui complique beaucoup la tâche des « services ».

PGP sera la cible principale de l'ire des services gouvernementaux, non parce qu'il serait un cryptosystème révolutionnaire, mais parce qu'il constitue une trousse à outils facile d'emploi pour l'usage quotidien, avec les outils de chiffrement symétrique et asymétrique, la gestion de « trousseaux de clés » publiques et privées, l'incorporation automatique de ces outils au logiciel de courrier électronique de l'utilisateur, sans oublier les accessoires de signature électronique. Zimmerman a aussi réalisé un excellent travail d'optimisation des algorithmes afin qu'un simple PC bas de gamme puisse effectuer les calculs cryptographiques à une vitesse raisonnable. Bref, on installe PGP (ou maintenant sa version libre GnuPG) sur son ordinateur personnel et ensuite tous les messages sont chiffrés et déchiffrés sans que l'on ait à s'en préoccuper. Les services semblaient mal supporter cette situation.

On trouvera sur le site `Lea-Linux.org` une bonne introduction pratique en français à GnuPG [21].

S/MIME et PGP : deux standards pour une messagerie sécurisée

La mise en œuvre du chiffrement et de la signature électronique lors d'échanges de courriers électroniques se fait en utilisant des standards de mise en forme spécifiques. Aux nombreux formats propriétaires d'applications spécifiques, s'ajoutent deux formats ouverts et plus répandus :

- S/MIME (RFC 2311) décrit le mode opératoire et d'encodage des courriers électroniques signés ou chiffrés avec utilisation de certificats X.509 et des clés privées associées ;
- PGP, historiquement plus ancien, utilise des principes similaires pour définir la manière dont il faut encoder un message afin d'y adjoindre la signature électronique, et la manière de représenter un message chiffré.

La différence essentielle entre les deux est le mode de certification de l'identité : alors que le système des anneaux PGP repose sur le principe « les amis de mes amis sont (peut-être) mes amis », les certificats X.509 utilisés dans S/MIME reposent, eux, sur un système de vérification hiérarchique (« l'autorité » a émis le certificat personnel et garantit l'identité selon des critères donnés).

S/MIME semble s'imposer comme la méthode la plus répandue : il est disponible avec de nombreux outils de messagerie du marché et peut être utilisé dès que l'utilisateur dispose d'un certificat personnel. La mise en œuvre de PGP pourra, elle, nécessiter l'installation

d'un programme complémentaire, opération également fort simple. Le choix de l'un ou de l'autre standard relève moins de choix techniques que de questions de goût, et de commodité d'usage avec tel ou tel logiciel de messagerie. Il est cependant admis qu'aujourd'hui PGP touche plutôt un public averti.

Le principe et les bases de fonctionnement des deux technologies sont similaires. Le message à signer ou à chiffrer (ou les deux) va être remis en forme dans un format spécifique avant d'être envoyé par le protocole SMTP aux destinataires. Ce qu'il faut comprendre c'est qu'avec de tels systèmes un message ayant plusieurs destinataires n'est ni signé ni chiffré plusieurs fois (dans le cas le plus courant bien sûr). La signature (le « résumé » du message chiffré avec la clé privée de l'expéditeur) est calculée une seule fois quel que soit le nombre de destinataires du message. Le standard permet de mettre plusieurs blocs de signatures dans un message, lorsque celui-ci est signé par plusieurs personnes, situation assez rare et qui n'est pas forcément prise en charge par les outils de messagerie les plus courants. Le chiffrement s'effectue également en plusieurs phases :

- en premier lieu, le message est chiffré avec un algorithme symétrique et une clé secrète créée pour l'occasion ;
- c'est cette clé secrète qui servira à créer un bloc de chiffrement pour chaque destinataire - la clé de l'algorithme symétrique est protégée par l'algorithme asymétrique en utilisant la clé publique de chaque destinataire, il y a donc autant de blocs « clé symétrique protégée » que de destinataires au message.

Pour émettre un message chiffré, il faut donc que l'expéditeur dispose de l'ensemble des clés publiques (PGP) ou certificats X.509 (S/MIME) de ses correspondants. Pour ne pas dupliquer le chiffrement du message il convient également que tous les destinataires sachent utiliser un même mode de chiffrement symétrique (par exemple le triple DES).

Créer un réseau de confiance

Du trousseau de clés à l'IGC

À ce stade de l'exposé, nous disposons de deux types de cryptosystèmes, l'un symétrique à secret partagé, l'autre asymétrique avec clés publiques et clés privées, le second permettant l'échange du secret partagé nécessaire au premier. Nous avons aussi, sans coût supplémentaire, un système de signature sûre et non répudiable qui garantit en outre l'intégrité des messages reçus. Ce qu'un système technique ne peut fournir à lui seul, c'est l'établissement du circuit de la confiance : comment être sûr que telle clé publique ne m'a pas été fournie par un usurpateur ?

PGP fournit à ce problème une solution à l'échelle d'une personne et de son cercle de relations : trousseau de clés publiques et privées conservé sur le disque dur d'un

ordinateur personnel. L'auteur de ces lignes emploie à cet usage l'excellent logiciel *KGpg* de la suite Kde. Les utilisateurs du logiciel de messagerie *Thunderbird* peuvent avoir recours à l'extension *Enigmail*. Il existe sur l'Internet des serveurs de clés, par exemple `http://subkeys.pgp.net/`, où l'on peut publier sa clé. Chacun peut signer les clés publiques des gens qu'il connaît, après avoir vérifié auprès de chacun, *de visu* et simultanément, une pièce d'identité et le condensat de la clé (cf. page 47) au cours d'une *séance de signature de clés (key signing party)*. Si je dois utiliser la clé publique de quelqu'un, le fait qu'elle soit signée par plusieurs personnes que je connais et dont j'ai moi-même vérifié la clé peut me convaincre de lui accorder ma confiance. Cette idée de *signer les clés* est cruciale dans le processus d'établissement de la confiance.

Mais il est patent que PGP ne répond pas, du moins à lui tout seul, à ce problème à l'échelle d'une entreprise, ni *a fortiori* à celle de l'Internet. Dès que le nombre de personnes concernées dépasse l'effectif d'un groupe d'amis, il faut penser à des solutions plus administrées pour engendrer une *transitivité de la confiance*.

Pour ce faire il faut recourir à un système d'annuaire électronique complété par une infrastructure de gestion de clés (IGC, en anglais *Public Key Infrastructure, PKI*), ce qui sera l'objet de la section suivante.

Annuaire électronique et gestion de clés

L'annuaire électronique est une base de données au format un peu particulier qui rend les services habituels d'un annuaire : répertorier des personnes ou des serveurs selon un schéma hiérarchique, de l'entité la plus englobante (pays) à la plus petite (personne) en passant par plusieurs niveaux (entreprise, département, service...). L'annuaire électronique contient aussi, idéalement, des certificats, qui comprennent notamment les clés publiques des entités enregistrées. Pour attester la véracité de ces certificats, ils sont, toujours idéalement, signés par une ou plusieurs autorités de certification, et éventuellement par le détenteur de la clé lui-même.

Il existe une norme assez généralement acceptée pour la structure hiérarchique de désignation des objets de l'annuaire, héritée de la norme d'annuaire X500 de l'ISO et adaptée de façon simplifiée par l'IETF pour les protocoles de l'Internet, sous le nom LDAP *(Lightweight Directory Access Protocol)*. La syntaxe ne fera peut-être pas l'unanimité, mais elle permet de traiter à peu près tous les cas possibles.

Voici le DN *(Distinguished Name)* de l'objet « Pierre Martin », c'est-à-dire son nom absolu, constitué de RDNs *(Relative Distinguished Names)* successifs, un peu comme les noms relatifs dans une arborescence de fichiers Unix constituent le chemin absolu d'un fichier ; CN signifie *Common Name*, OU *Organizational Unit*, O *Organization* :

```
cn=Pierre Martin, ou=Groupe Système,
ou=Division Informatique, o= Compagnie Dupont
```

La forme des certificats découle également de la norme X500, et elle obéit à la norme X509.

Qui certifie la signature des autorités de certification ? En bref, qui me garantit que le contenu de l'annuaire n'est pas un artefact créé par un escroc ? La procédure de création de l'IGC et d'enregistrement des entités comportera nécessairement une intervention humaine qui à chaque étape constate l'identité de la personne (physique, morale ou technique) à laquelle est délivré le certificat. Un certificat émis par l'IGC décrit une entité et contient sa clé publique, ainsi que les signatures des autorités qui garantissent le certificat.

Dans un monde idéal (idéal du point de vue de la sécurité informatique, qui n'est certes pas le seul envisageable), une hiérarchie d'IGC propage une certaine confiance. Quiconque accède à un système d'information est identifié par l'annuaire et authentifié par son certificat et sa clé publique, dont le pendant est la clé privée. Chaque serveur est également identifié et authentifié. Les communications entre les deux peuvent être chiffrées.

Risques liés aux systèmes d'identification

Le fonctionnement des espaces d'identifiants universels évoqués ci-dessus en référence à l'article de Sophie Le Pallec [72] est crucial dans le monde contemporain : une interruption de plusieurs heures des services rendus par l'Internet aurait des conséquences économiques et organisationnelles considérables.

L'*Internet Corporation for Assigned Names and Numbers (ICANN)*, nous l'avons signalé ci-dessus, a notamment pour mission de gérer le système de noms de domaine de premier niveau pour les codes génériques (gTLD, tels `.com` ou `.edu`)

et les codes nationaux (ccTLD, tels `.fr` ou `.be`), et d'assurer l'organisation et le fonctionnement du système de serveurs racine : si ce système de serveurs racine est indisponible, l'Internet est arrêté. On imagine qu'une attaque réussie contre ce système serait pour les pirates qui l'auraient entreprise un « succès » de première grandeur.

Une telle attaque contre les serveurs racine ne serait pas une chose facile : aujourd'hui, ces serveurs sont au nombre de 13, répartis à la surface de la planète, surtout aux États-Unis, mais aussi à Tokyo, Londres et Stockholm, et chacun de ces serveurs est lui-même répliqué sur plusieurs machines. En outre, comme ces machines sont souvent attaquées, leurs administrateurs sont bien entraînés et ils sont au fait des dernières techniques malfaisantes.

Un collègue avec qui j'évoquais l'éventualité d'une telle attaque me faisait remarquer qu'une attaque contre l'entreprise Verisign serait plus habile : Verisign gère le domaine `.com`, et occupe une position éminente sur le marché des certificats électroniques à usage commercial. En effet, le certificat de l'Autorité de Certification racine de Verisign figure dans tous les navigateurs, ce qui fait que les certificats vendus par Verisign sont automatiquement acceptés lors de transactions électroniques, alors qu'un certificat délivré par une IGC moins reconnue n'est accepté que si l'utilisateur charge au préalable dans son navigateur le certificat de l'Autorité de Certification racine concernée, manœuvre simple mais qui suffit à dissuader le client.

La mise hors service des infrastructures de Verisign ne suffirait pas à invalider les certificats en circulation sur le réseau, puisqu'ils sont déjà enregistrés dans les navigateurs : pour obtenir un résultat qui s'approche de cet idéal, le meilleur moyen serait sans doute de diffuser un *Service Pack* falsifié modifiant les magasins de certificats des utilisateurs qui le mettraient en service.

En revanche, une attaque couronnée de succès sur Verisign aurait des effets très néfastes sur le fonctionnement du domaine `.com` et, partant, sur le commerce électronique mondial, dont le volume est estimé à 3% du PNB mondial consolidé, ce qui est considérable : il en résulterait sûrement des dommages économiques non négligeables.

Organiser un système d'identité numérique

Objectif SSO

La prolifération des systèmes d'identification et d'authentification est une conséquence non désirée de la pénétration de l'informatique dans tous les domaines de l'activité humaine.

L'utilisation de certificats électroniques archivés dans des annuaires aurait pour avantage, outre de faire obstacle plus efficacement à la fraude informatique, de permettre aux personnes de posséder un système d'identification électronique unique *(single sign on)* au lieu d'avoir à connaître des dizaines de mots de passe et codes secrets — pour leur carte bancaire, leur téléphone mobile, leurs courriers électroniques privé et professionnel, la consultation en ligne de leurs comptes bancaires, les différents systèmes informatiques auxquels elles accèdent dans leur vie professionnelle et privée.

Expérience de terrain

L'auteur de ces lignes a eu un jour à lancer et à réaliser un projet d'annuaire électronique dans un grand organisme de recherche scientifique. Il apparut assez vite que ce type de projet était notablement plus complexe qu'il n'y paraît lorsque l'on n'a pas essayé.

Constituer l'annuaire papier, donc statique, d'une population donnée consiste à effectuer à un instant donné un recensement exhaustif des individus de cette population et de leurs caractéristiques qui doivent figurer dans l'annuaire, telles qu'adresse, numéro de téléphone, etc.

Pour constituer un annuaire électronique qui présente des avantages significatifs par rapport à un annuaire papier, il faut créer une base de données qui contienne les données déjà évoquées ci-dessus, et surtout mettre en place des processus d'alimentation et de mise à jour de cette base avec pour objectifs les qualités suivantes : pertinence, actualité, fiabilité, exhaustivité, disponibilité. Concevoir ces processus d'alimentation de l'annuaire demande d'avoir identifié les sources adéquates de données.

La réponse naïve à cette question, qui fut donnée par quelques aspirants-prestataires, va de soi : « Eh bien, vous prenez votre fichier de personnel, une extraction, et voilà ! » C'était simplement oublier qu'un organisme de recherche

réunit bien d'autres individus que ses propres personnels, lesquels ne représentent qu'une petite moitié des quelques milliers de personnes actives dans l'institution. Pour écarter d'autres idées simplistes, qu'il suffise de dire que lesdits personnels sont dispersés sur quelques dizaines de sites dotés chacun de ses propres règles de gestion administrative et technique. Il fallait chercher autre chose.

Une enquête auprès de collègues expérimentés m'apprit qu'il existait environ 130 fichiers de personnel dans l'entreprise, exhaustifs ou partiels. De quoi faire dresser les cheveux sur la tête d'un concepteur de système d'information ! Il est plus que probable que l'existence de bases de données de bonne qualité facilement accessibles serait de nature à faire disparaître beaucoup de ces fichiers d'intérêt local, constitués pour résoudre des problèmes ponctuels, et pas forcément toujours de très bonne qualité. Mais sans faire passer le nombre de fichiers de personnel de 130 à 1 !

Une visite aux détenteurs de quelques-uns des 130 fichiers a révélé un certain nombre de fichiers redondants, morts ou indigents, mais aussi des fichiers bien vivants qui avaient de bonnes raisons de continuer à mener une existence distincte. Parmi les bonnes raisons, la plupart ont trait à des exigences temporelles quant à la disponibilité des données. Ainsi, lorsqu'un nouveau salarié prend ses fonctions le premier jour du mois, le degré d'urgence de son inscription dans les fichiers du personnel est déterminé par l'objectif de lui verser sa rémunération, c'est-à-dire que cette inscription doit avoir lieu au plus tard entre le 15 et le 20 du mois. Mais pour qu'il puisse effectivement commencer à travailler il faut lui ouvrir un compte de messagerie électronique bien avant cette date, et, pour ce faire, l'enregistrer dans les bases de données correspondantes. Il serait également souhaitable qu'il soit dans l'annuaire téléphonique. Bref, les auteurs et les utilisateurs de ces fichiers ont des impératifs différents, sans même aborder la délicate question de la confidentialité des données et du secret professionnel. Les supprimer au profit d'une base de données unique n'irait sûrement pas sans poser de difficiles problèmes.

Nos dernières illusions s'envolèrent lorsque nous décidâmes de quitter le havre de la direction générale pour visiter des sites opérationnels en province ou en région parisienne. Les personnes chargées d'alimenter les bases en données recueillies sur le terrain nous expliquèrent avec ménagement mais franchise les procédures suivies. Les fichiers dont le contenu avait une incidence financière ou en termes de personnel étaient mis à jour sérieusement, mais uniquement pour les données

qui avaient une telle incidence. D'autres fichiers à usage purement bureaucratique (ou perçu comme tel), et dont les procédures de mise à jour étaient de surcroît particulièrement pénibles, étaient beaucoup moins bien traités — en général on se contentait de renvoyer la version de l'année précédente, et personne ne s'apercevait de rien.

Bref, nous avions rêvé d'un système d'information où il nous aurait suffi de voleter de base de données en base de données pour y butiner les données utiles à notre projet : il se révélait que nous avions à construire les données dont nous avions besoin, et que la réutilisation de données existantes n'était pas un avantage mais bien plutôt une contrainte, imposée par le souci de cohérence mais assortie d'un coût élevé induit par leur mauvaise qualité et, paradoxalement, par leur incohérence.

La situation décrite ci-dessus ne correspond pas à un cas particulièrement défavorable : au contraire, tous les univers de données réels sont peu ou prou conformes à cette description. Les données sont bonnes si elles ont une bonne raison de l'être, et elles sont bonnes à l'usage pour lequel elles ont été construites. Si l'on veut en faire autre chose, il faut les ré-élaborer entièrement.

Troisième partie

Politiques de sécurité du système d'information

8

Une charte des utilisateurs

Ce chapitre est consacré à un exemple de charte qu'un organisme de recherche doit faire signer aux utilisateurs de son système d'information. Un tel document doit être approuvé par les organismes de concertation entre le personnel et la direction de l'établissement, c'est-à-dire en droit français le Comité d'entreprise pour les organismes de droit privé et le Comité technique paritaire central pour les organismes publics, puis il doit être promulgué par la direction au plus haut niveau. Une fois que ces formalités ont été accomplies, la charte devient partie intégrante du règlement intérieur de l'entreprise, et peut donc être opposée aux membres du personnel qui la transgressent, y compris devant les instances disciplinaires et juridiques. En outre, un membre du personnel qui enfreindrait une loi pénale signalée par la charte ne pourrait pas arguer de sa bonne foi devant un tribunal, ni rejeter la responsabilité de ses actes délictueux sur son employeur.

Une telle charte est destinée à faire l'objet d'une large publicité, et notamment à paraître sur le site Web de l'organisme.

Préambule de la charte

Cette charte de l'utilisateur des ressources informatiques et des services Internet de l'INSIGU[1] est avant tout un code de bonne conduite. Il a pour objet de préciser la responsabilité des utilisateurs en accord avec la législation afin d'instaurer un usage convenable des ressources informatiques et des services Internet, dans le respect des dispositions légales et réglementaires en vigueur, avec des règles minimales de courtoisie et de respect d'autrui.

Pour tout renseignement complémentaire, les utilisateurs peuvent s'adresser, selon le cas, au responsable de leur unité, équipe, département ou service, au responsable régional informatique de la direction régionale dont ils dépendent, ou au responsable de la sécurité des systèmes d'information de l'INSIGU.

Définitions

On désignera sous le terme « entité » les structures créées par l'INSIGU pour l'accomplissement de ses missions, telles que les unités de recherche, les équipes, ainsi que les départements et services administratifs.

On désignera de façon générale sous le terme « ressources informatiques », les moyens informatiques de calcul ou de gestion locaux ainsi que ceux auxquels il est possible d'accéder à distance, directement ou en cascade à partir du réseau administré par une entité de l'INSIGU.

On désignera par « services Internet », la mise à disposition par des serveurs locaux ou distants de moyens d'échanges et d'informations diverses : Web, messagerie, forum...

On désignera sous le terme « utilisateur », les personnes ayant accès aux ressources informatiques et services Internet d'une entité de l'INSIGU.

[1] L'Institut national des sciences informatiques et géographiques de l'univers (INSIGU) est un organisme de recherche fictif, pour les besoins de notre exemple.

Accès aux ressources et aux services

L'utilisation des ressources informatiques et l'usage des services Internet ainsi que du réseau pour y accéder sont autorisés dans le cadre exclusif de l'activité professionnelle des utilisateurs conformément à la législation en vigueur.

L'activité professionnelle est celle prévue par les statuts du réseau MIRANDA pour la recherche scientifique, auquel est lié l'INSIGU, à savoir : les activités de recherche, d'enseignement, de développement technique, de transfert de technologies, de diffusion d'informations scientifiques, techniques et culturelles, d'expérimentation de nouveaux services présentant un caractère d'innovation technique, mais également toute activité administrative et de gestion découlant de ces activités ou les accompagnant.

L'utilisation des ressources informatiques partagées de l'entité et la connexion d'un équipement sur le réseau sont en outre soumises à autorisation. Ces autorisations, délivrées par le directeur de l'entité, sont strictement personnelles et ne peuvent en aucun cas être cédées, même temporairement, à un tiers. Ces autorisations peuvent être retirées à tout moment. Toute autorisation prend fin lors de la cessation, même provisoire, de l'activité professionnelle qui l'a justifiée.

L'entité pourra en outre prévoir des restrictions d'accès spécifiques à son organisation : chiffrement d'accès ou d'authentification, filtrage d'accès sécurisé, etc.

Règles d'utilisation, de sécurité et de bon usage

Tout utilisateur est responsable de son usage des ressources informatiques et du réseau auxquels il a accès. Il a aussi la charge, à son niveau, de contribuer à la sécurité générale et aussi à celle de son entité.

L'utilisation de ces ressources doit être rationnelle et honnête afin d'en éviter la saturation ou le détournement à des fins personnelles.

En particulier :

- il doit appliquer les recommandations de sécurité de l'entité à laquelle il appartient ;
- il doit assurer la protection de ses informations et il est responsable des droits qu'il donne éventuellement à d'autres utilisateurs, il lui appartient

de protéger ses données en utilisant les différents moyens de sauvegarde individuels ou mis à sa disposition ;
- il doit signaler toute tentative de violation de son compte et, de façon générale, toute anomalie qu'il peut constater ;
- il doit suivre les règles en vigueur au sein de l'entité pour toute installation de logiciel ;
- il choisit des mots de passe sûrs, gardés secrets et il ne doit en aucun cas les communiquer à des tiers ;
- il s'engage à ne pas mettre à la disposition d'utilisateurs non autorisés un accès aux systèmes ou aux réseaux, à travers des matériels dont il a l'usage ;
- il ne doit pas utiliser ou essayer d'utiliser des comptes autres que le sien, ni tenter de masquer sa véritable identité ;
- il ne doit pas tenter, directement ou indirectement, de lire, modifier, copier ou détruire des données autres que celles qui lui appartiennent en propre ; en particulier, il ne doit pas modifier le ou les fichiers contenant des informations comptables ou d'identification ;
- il ne doit pas quitter son poste de travail ni ceux en libre-service en laissant des ressources ou services accessibles et il doit se déconnecter, sauf avis contraire de l'administrateur du réseau.

Confidentialité

L'accès par les utilisateurs aux informations et documents conservés sur les systèmes informatiques doit être limité à ceux qui leur sont propres, et ceux qui sont publics ou partagés. En particulier, il est interdit de prendre connaissance d'informations détenues par d'autres utilisateurs, quand bien même ceux-ci ne les auraient pas explicitement protégées. Cette règle s'applique également aux conversations privées de type courrier électronique dont l'utilisateur n'est destinataire ni directement ni en copie. Si, dans l'accomplissement de son travail, l'utilisateur est amené à constituer des fichiers relevant de la loi Informatique et Libertés, il devra auparavant en avoir fait la demande à la CNIL en concertation avec le Directeur de l'entité, le correspondant informatique et libertés de l'INSIGU et le service juridique de l'INSIGU et en avoir reçu l'autorisation. Il est rappelé que cette au-

torisation n'est valable que pour le traitement défini dans la demande et non pour le fichier lui-même.

Respect de la législation

Il est strictement interdit d'effectuer des copies de logiciels commerciaux pour quelque usage que ce soit, hormis une copie de sauvegarde dans les conditions prévues par le code de la propriété intellectuelle. Ces dernières ne peuvent être effectuées que par la personne habilitée à cette fin par le responsable de l'entité.

Par ailleurs l'utilisateur ne doit pas installer de logiciels à caractère ludique, ni contourner les restrictions d'utilisation d'un logiciel.

Il est rappelé que les logiciels commerciaux disponibles pour les utilisateurs de l'INSIGU sont l'objet de licences par lesquelles des droits d'usage sont concédés à l'INSIGU. Ces licences font l'objet de contrats conclus par l'INSIGU. Il est de la responsabilité des personnels de respecter les termes de ces licences et de ces contrats ; y manquer serait un délit et, en outre, une faute professionnelle.

De même, l'installation sur un système informatique mis en œuvre par l'INSIGU d'un logiciel dont le droit d'usage est acquis à titre privé par un membre du personnel n'est pas autorisée.

L'usage de logiciels commerciaux est régi par des contrats et protégé par des lois qui entraînent une responsabilité personnelle de leur utilisateur, que la responsabilité propre de l'INSIGU en tant que personne morale ne saurait exonérer.

Préservation de l'intégrité des systèmes informatiques

L'utilisateur s'engage à ne pas apporter volontairement de perturbations au bon fonctionnement des systèmes informatiques et des réseaux (internes ou extérieurs à l'INSIGU), que ce soit par des manipulations anormales du matériel, ou par l'introduction de logiciels parasites connus sous le nom générique de virus, chevaux de Troie, bombes logiques...

Tout travail de recherche ou autre risquant de conduire à la violation de la règle définie au paragraphe précédent ne pourra être accompli qu'avec l'autorisation du responsable de l'entité et dans le strict respect des règles qui auront alors été définies.

Il est de la responsabilité de l'utilisateur de s'assurer de l'installation sur l'ordinateur qu'il utilise régulièrement de logiciels de protection contre les logiciels parasites évoqués ci-dessus. Le département du système d'information organise la distribution des logiciels de protection appropriés.

Usage des services Internet (Web, messagerie, forum...)

L'utilisateur doit faire usage des services Internet dans le cadre exclusif de ses activités professionnelles et dans le respect de principes généraux et des règles propres aux divers sites qui les proposent ainsi que dans le respect de la législation en vigueur.

En particulier il doit respecter les règles suivantes.

Règles de bon usage

- il ne doit pas se connecter ou essayer de se connecter sur un serveur autrement que par les dispositions prévues par ce serveur ou sans y être autorisé par les responsables habilités ;
- il ne doit pas se livrer à des actions mettant sciemment en péril la sécurité ou le bon fonctionnement des serveurs auxquels il accède ;
- il ne doit pas usurper l'identité d'une autre personne et il ne doit pas intercepter de communications entre tiers ;
- il ne doit pas utiliser ces services pour proposer ou rendre accessibles aux tiers des données et informations confidentielles ou contraires à la législation en vigueur ;
- il ne doit pas déposer des documents sur un serveur sauf si celui-ci le permet, ou sans y être autorisé par les responsables habilités ;
- il doit faire preuve de la plus grande correction à l'égard de ses interlocuteurs dans les échanges électroniques par courrier, forums de discussions...

- il n'émettra pas d'opinions personnelles étrangères à son activité professionnelle susceptibles de porter préjudice à l'INSIGU ou à ses agents ;
- il doit respecter les lois et notamment celles relatives aux publications à caractère injurieux, raciste, pornographique ou diffamatoire.

Publication sur l'Internet

La mise à la disposition du public d'un serveur Web appartenant au domaine `insigu.fr`, ou affichant le logo de l'INSIGU ou manifestant de toute autre façon son appartenance à l'INSIGU engage la responsabilité de l'INSIGU et expose son image. L'ouverture d'un tel site est donc soumise à l'autorisation du département de l'information scientifique et de la communication. La publication de documents sur un site autorisé se fera ensuite sous la responsabilité des responsables d'entité, sous le contrôle *a posteriori* du département de l'information scientifique et de la communication, et selon les principes énoncés par la charte de bonne utilisation du réseau Internet dans les laboratoires INSIGU, disponible sur le serveur `http://www.insigu.fr`.

Responsabilité légale

La publication d'informations et de documents sur un support public tel que le Web entraîne une responsabilité personnelle de leur auteur devant la loi, que la responsabilité de l'INSIGU en tant que personne morale ne saurait exonérer.

Dispositifs de filtrage de trafic

L'INSIGU met en oeuvre des dispositifs de contrôle du trafic provenant de l'Internet. Il s'agit notamment d'un système obligatoire de mandataires (proxy) effectuant un contrôle antivirus sur les documents chargés ainsi que d'un système de filtrage d'URL destiné à interdire l'accès à certains sites ou certains types de documents.

Toute l'activité de navigation, les accès autorisés ou interdits sont enregistrés et conservés par l'INSIGU pour une durée d'un an, conformément à la législation.

La mise en œuvre de cette solution est faite dans le respect de la législation et a donné lieu à une information préalable des instances représentatives du personnel et du comité d'entreprise.

Surveillance et contrôle de l'utilisation des ressources

Pour des nécessités de maintenance et de gestion technique, l'utilisation des ressources matérielles ou logicielles ainsi que les échanges via le réseau peuvent être analysés et contrôlés dans le respect de la législation applicable et notamment de la loi sur l'informatique et les libertés.

La mise en œuvre de ces mesures est faite dans le respect de la législation et a donné lieu à une information préalable des instances représentatives du personnel et du comité d'entreprise.

Rappel des principales lois françaises :

Il est rappelé que toute personne présente sur le sol français doit respecter la législation française, notamment dans le domaine de la sécurité informatique :

- la loi du 6/1/78 dite « informatique et liberté », (cf. le site web de la CNIL `http://www.cnil.fr/`);
- la législation relative à la fraude informatique, (article 323-1 à 323-7 du Code pénal),(cf. `http://www.legifrance.gouv.fr/`)
- la législation relative à la propriété intellectuelle (cf. `http://www.legifrance.gouv.fr/`);
- la loi du 04/08/1994 relative à l'emploi de la langue française, (cf. `http://www.culture.fr/culture/dglf/`);
- la législation applicable en matière de cryptologie, (cf. `http://www.ssi.gouv.fr/fr/reglementation/index.html`).

Application

La présente charte s'applique à l'ensemble des agents de l'INSIGU tous statuts confondus, et plus généralement à l'ensemble des personnes utilisant, de façon permanente ou temporaire, les moyens informatiques de l'entité ainsi que ceux auxquels il est possible d'accéder à distance directement ou en cascade à partir du réseau administré par l'entité.

Elle sera annexée, à titre d'information, aux contrats de travail conclus avec les agents contractuels et vacataires qui auront accès au système informatique de leur entité.

Elle sera en outre signée par toutes personnes accueillies à l'INSIGU et ayant accès audit système.

9

Une charte de l'administrateur système et réseau

La multiplication de questions de plus en plus complexes liées à la sécurité des systèmes et des réseaux, l'imbrication de plus en plus intime des aspects techniques et juridiques de ces questions et le risque accru de conséquences judiciaires en cas d'erreur incitent à la rédaction, au sein de chaque entreprise ou organisation, d'une *charte de l'administrateur de système et de réseau* qui rappelle les devoirs, les pouvoirs et les droits des ingénieurs et des techniciens qui administrent la sécurité des réseaux, des ordinateurs et en fin de compte du système d'information. Le présent chapitre énonce les principes qui peuvent conduire la rédaction d'un tel document, puis en propose un exemple pour une entreprise fictive.

Complexité en expansion et multiplication des risques

L'activité de l'administrateur de système et de réseau le confronte à un certain nombre de paradoxes : par exemple, il doit configurer son système d'acheminement de messagerie électronique (*Mail Transfer Agent, MTA*, ou passerelle de messagerie) de façon à tenir un journal de tous les messages émis et reçus par le point d'accès à l'Internet dont il est responsable, c'est une obligation légale. Mais s'il oublie de détruire ces journaux à l'issue d'un délai maximal d'un an, il enfreint une autre obligation légale qui résulte des directives de la CNIL.

Cette activité d'administration de la passerelle de messagerie de l'entreprise lui permet de détecter les usages contraires à la loi qui pourraient en être faits par des employés indélicats, dont les exemples les plus courants sont, non limitativement :

- envoi de messages ou abonnement à des listes de diffusion susceptibles de tomber sous le coup des lois qui répriment le racisme et la xénophobie, la pédophilie ou le trafic d'êtres humains ;
- communication à des tiers d'informations couvertes par le secret professionnel, qui constituent le patrimoine intellectuel de l'entreprise, et dont la divulgation à des concurrents est de nature à causer un préjudice certain ;
- infraction à la législation sur la propriété littéraire et artistique, lorsque les serveurs de l'entreprise sont utilisés pour télécharger ou, pire, redistribuer des œuvres musicales ou cinématographiques couvertes par des droits d'auteur ;
- délit de presse, par l'ouverture de sites Web ou de forums au contenu susceptible d'être attaqué au titre des lois sur la diffamation, le plagiat, etc.

La constatation de telles infractions lui fait devoir d'y mettre fin, mais dans les cas où les manifestations de ces actes ne sont pas publiques (cas du courrier électronique), s'il en fait état dans un rapport à la direction de l'entreprise, il s'expose à être condamné par un tribunal en vertu de la loi qui protège le secret de la correspondance. En effet, si la jurisprudence (arrêt du 17 décembre 2001 de la Cour d'appel de Paris, « ESPCI », École Supérieure de Physique et Chimie industrielle) reconnaît que l'administrateur détient la possibilité technique de lire les contenus des messages, celui-ci n'est en revanche pas autorisé à les divulguer même à ses supérieurs hiérarchiques.

« Ainsi la délicate mission de l'administrateur sera de mettre fin au comportement frauduleux ou préjudiciable sans en informer son supérieur hiérarchique qui dispose pourtant de l'autorité et du pouvoir de décision », note Laurence Freyt-Caffin [56].

De façon plus générale, l'administrateur de système et de réseau a accès à toutes les données de l'entreprise et des utilisateurs qui stationnent ou circulent sur les machines et les réseaux dont il a la responsabilité : ce pouvoir le soumet en permanence à la tentation d'en abuser, même si ce n'est que pour simplifier sa tâche, ou rendre service aux utilisateurs, ou pour assurer le bon fonctionnement des infrastructures en question.

De façon nettement plus embarrassante, il peut recevoir de sa hiérarchie des injonctions contraires aux lois : il est alors placé devant le dilemme d'avoir à désobéir à ces injonctions, ce qui peut mettre en péril sa situation professionnelle, ou d'enfreindre la loi, ce qui risque de le mener devant un juge.

Règles de conduite

L'administrateur de systèmes et de réseaux dispose de pouvoirs importants : il importe de circonscrire avec soin l'usage qu'il peut en faire afin d'éviter les abus, notamment par l'atteinte à la confidentialité des échanges et des données.

Secret professionnel

Le devoir de secret professionnel s'impose aux administrateurs ayant accès aux données personnelles des utilisateurs dans le cadre de leurs fonctions.

1. Arrêt de la Chambre sociale de la Cour de cassation en date du 2 octobre 2001 : « Attendu que le salarié a droit, même au temps et au lieu de travail, au respect de l'intimité de sa vie privée ; que celle-ci implique en particulier le secret des correspondances ; que l'employeur ne peut dès lors sans violation de cette liberté fondamentale prendre connaissance des messages personnels émis par le salarié et reçus par lui grâce à un outil informatique mis à sa disposition pour son travail et ceci même au cas où l'employeur aurait interdit une utilisation non professionnelle de l'ordinateur. »

2. Code du Travail, articles L432-2-1 : « Le comité d'entreprise est informé, préalablement à leur utilisation, sur les méthodes ou techniques d'aide au recrutement des candidats à un emploi ainsi que sur toute modification de celles-ci. Il est aussi informé, préalablement à leur introduction dans l'entreprise, sur les traitements automatisés de gestion du personnel et sur toute modification de ceux-ci. Le comité d'entreprise est informé et consulté, préalablement à la décision de mise en œuvre dans l'entreprise, sur les moyens ou les techniques permettant un contrôle de l'activité des salariés. »

Mots de passe

J'emprunte ici à Patrick Chambet les idées qu'il a exprimées sur la liste de diffusion de l'Ossir[1] :

« — Non ! Les administrateurs ne doivent jamais connaître les mots de passe des utilisateurs.

— Pourquoi l'administrateur n'a-t-il pas besoin de connaître les mots de passe ?

— Un administrateur est, par définition, celui qui possède des privilèges élevés. En particulier, il peut effectuer toutes les tâches nécessaires en l'absence des utilisateurs, comme par exemple la prise de possession de fichiers, la modification des permissions d'accès à des ressources, etc. Pour cela, il n'a pas besoin et ne doit pas [commettre d'usurpation de personnalité] (se loger avec le *login* et le mot de passe de l'utilisateur).

S'il doit tout de même se résoudre à cela, l'utilisateur légitime devrait être présent (ce point devrait être débattu par les juristes de la liste, car le règlement intérieur de l'entreprise, la charte informatique, la politique de sécurité et les lois, décrets et jurisprudence entrent en jeu).

Il arrive que l'administrateur fasse tourner un *craqueur* de mots de passe pour vérifier la robustesse des mots de passe des utilisateurs. Mais dans ce cas, dès qu'un mot de passe est craqué, il doit demander immédiatement à l'utilisateur de le changer pour un mot de passe de robustesse au moins équivalente. Il ne le connaît donc plus.

— Pourquoi l'administrateur ne doit-il pas connaître les mots de passe ?

[1] `http://www.ossir.org`

— Tout d'abord pour dégager sa responsabilité en cas d'activité délictueuse effectuée à l'aide d'un compte utilisateur particulier : l'utilisateur en question ne pourra plus dire que ce n'est pas lui, mais l'administrateur qui a envoyé tel [message électronique].

Ensuite pour le respect de la confidentialité des ressources utilisateurs (classifiées ou non), même si, par définition, un administrateur pourra toujours, à l'aide d'une action volontaire et avec une intention évidente (plaidable devant un juge si l'administrateur n'a pas reçu d'ordre explicite), accéder aux ressources en question.

D'un côté l'administrateur est protégé, de l'autre il devient plus facilement condamnable. »

Les injonctions hiérarchiques à violer le secret des mots de passe sont fréquentes, souvent pour des raisons en apparence excellentes : accéder aux données cruciales détenues par un utilisateur en vacances et inaccessible en est l'exemple typique. Il peut être très difficile de résister à une telle demande, et l'utilisateur à son retour peut détecter l'intrusion en consultant les journaux du système. Certes l'administrateur peut détruire ou modifier les éléments de journalisation relatifs à son action, mais cette altération même des journaux peut être détectée, quoique plus difficilement, et en cas de comparution devant un tribunal il aura ainsi considérablement aggravé sa faute. Si pour une raison ou pour une autre les relations entre le possesseur des données et son employeur ou l'administrateur sont conflictuelles, on voit toutes les conséquences fâcheuses que peut entraîner cet enchaînement de circonstances. Il convient donc d'éviter absolument de commettre de telles actions.

Proposition de charte

La présente charte de l'administrateur de système et de réseau de l'INSIGU[2] est destinée à déterminer les devoirs, les pouvoirs et les droits des ingénieurs et des techniciens qui administrent la sécurité des réseaux, des ordinateurs et du système d'information de l'INSIGU.

[2] L'Institut national des sciences informatiques et géographiques de l'univers (INSIGU) est un organisme de recherche fictif, pour les besoins de notre exemple.

Cette charte est promulguée en référence à la charte de l'utilisateur des ressources informatiques et des services Internet de l'INSIGU (cf. chapitre 8 page 185), qu'elle complète et dont elle est inséparable.

Définitions

Les *entités* de l'INSIGU, ses *ressources informatiques*, ses *services Internet* et les *utilisateurs* du système d'information qu'ils constituent sont définies ici comme dans la charte de l'utilisateur des ressources informatiques et des services Internet de l'INSIGU (cf. section 8 page 186).

L'*administrateur* d'un système ou d'un réseau de l'INSIGU est toute personne, employée ou non par l'INSIGU, à laquelle a été confiée explicitement et par écrit, sous la forme d'une lettre de mission, d'un profil de poste annexé au contrat de travail ou d'un contrat de prestations de service, la responsabilité d'un système informatique, d'un réseau ou d'un sous-réseau administrés par une entité de l'INSIGU, ou de plusieurs de ces éléments. Une personne à qui a été conférée une telle responsabilité sera désignée dans la suite de ce document par le terme *administrateur*. L'ensemble des éléments sur lesquels s'exerce cette responsabilité constitue le *périmètre d'activité* de l'administrateur.

Le *comité de coordination* de sécurité du système d'information (SSI) est constitué de responsables chargés d'émettre des règles et des recommandations dans le domaine SSI, de prendre les mesures appropriées pour qu'elles soient mises en vigueur, et d'organiser les activités de formation, d'information et de sensibilisation de nature à améliorer les conditions de leur application ; il est en outre chargé de suivre la juridiction et notamment les arrêtés et jurisprudences. Les membres de ce comité de coordination sont le Responsable de sécurité des systèmes d'information (RSSI) de l'INSIGU, le responsable de la sécurité opérationnelle au sein du département du système d'information (DSI) de l'INSIGU, le correspondant informatique et libertés de l'INSIGU et d'autres personnes désignées par le directeur général de l'INSIGU ou son représentant autorisé, notamment un représentant du département des affaires juridiques.

Les devoirs, les pouvoirs et les droits de l'administrateur, définis dans la présente charte, constituent ensemble les *responsabilités SSI* de l'administrateur.

Les consignes du comité de coordination SSI s'imposent aux administrateurs de systèmes et de réseaux pour l'exercice de leurs responsabilités SSI dans leur périmètre d'activité.

Responsabilités du comité de coordination SSI

Surveillance et audit

Le comité de coordination SSI organise la surveillance et l'audit de toutes les activités des systèmes et de tous les trafics réseau sur les infrastructures administrées par l'INSIGU.

Pour ce faire, le comité de coordination SSI est habilité à donner des consignes de surveillance, de recueil d'information et d'audit aux administrateurs concernés.

Contrôle d'accès

Le comité de coordination SSI définit des règles de contrôle d'accès aux systèmes et aux réseaux conformes à la présente charte et à la charte de l'utilisateur des ressources informatiques et des services Internet de l'INSIGU.

Vérification

Le comité de coordination SSI et les administrateurs concernés sont habilités à entreprendre toutes actions appropriées pour vérifier la bonne application des règles de contrôle d'accès aux systèmes et aux réseaux définies à l'article précédent, ainsi que pour détecter leurs vulnérabilités.

Responsabilités de l'administrateur de système et de réseau

Enregistrement des incidents de sécurité

L'administrateur conserve une trace écrite des incidents de sécurité survenus dans son périmètre d'activité. Cette trace doit comporter les indications de date et d'heure des événements considérés, et une description de ces événements.

Notification des incidents de sécurité

Les administrateurs de système et de réseau sont tenus de déclarer tout incident de sécurité au RSSI et au responsable de la sécurité opérationnelle. Les directives

du RSSI et du responsable de la sécurité opérationnelle pour des actions relatives aux incidents sont mises en application sans délais.

Journalisation et archivage

L'administrateur active sur les systèmes dont il a la responsabilité les journaux nécessaires à l'identification et à la reconstitution des séquences d'événements qui pourraient constituer un incident de sécurité, ou qui pourraient faire l'objet d'une commission rogatoire émise par les autorités judiciaires. Il archive les données ainsi recueillies dans des conditions propres à en assurer l'intégrité, la disponibilité, l'authenticité et la confidentialité.

Il mène cette activité de journalisation et d'archivage dans des conditions qui garantissent le respect des lois et des règlements relatifs aux libertés publiques et privées, au secret des correspondances, au droit d'accès à l'information, et il veille notamment à détruire tous les journaux qui comportent des données nominatives à l'expiration d'un délai qui ne peut excéder un an, ou le délai légal à la date considérée.

Parmi les textes législatifs et réglementaires qui s'appliquent à cette activité, il convient d'accorder une attention particulière à la norme simplifiée n° 46 de la Commission nationale informatique et libertés, « destinée à simplifier l'obligation de déclaration des traitements mis en œuvre par les organismes publics et privés pour la gestion de leurs personnels » [3].

Examen des journaux

L'administrateur examine régulièrement les journaux mentionnés à l'article ci-dessus.

Dérogations aux règles SSI

Les règles SSI mentionnées dans la présente charte, dans la charte de l'utilisateur des ressources informatiques et des services Internet de l'INSIGU, ou édictées par le RSSI de l'INSIGU, par le responsable de la sécurité opérationnelle au sein du DSI de l'INSIGU ou par le comité de coordination SSI s'imposent à tous les utilisateurs des systèmes d'information de l'INSIGU, qu'ils soient on non des

[3] `http://www.cnil.fr/index.php?id=1231`

employés de l'INSIGU. Les administrateurs de systèmes et de réseaux de l'INSIGU ont pour mission de les mettre en œuvre et de les faire respecter dans leur périmètre d'activité.

Les responsables d'entités qui voudraient passer outre ces règles SSI, ou entreprendre des actions qui dérogeraient à ces règles, doivent :

- remettre à l'administrateur responsable des infrastructures concernées un document écrit et signé par lequel ils assument explicitement la responsabilité de cette dérogation, des risques qui en découlent, et de leurs conséquences ;
- obtenir du directeur général de l'INSIGU ou de son représentant désigné une décharge écrite pour le RSSI, le DSI et affiliés.

Les utilisateurs qui ne seraient pas responsables d'entités et qui voudraient bénéficier de telles dérogations doivent obtenir qu'elles soient endossées par leur responsable d'entité, dans les conditions indiquées à l'alinéa précédent.

Identification des utilisateurs et contrôles d'accès

Dans leur périmètre d'activité, les administrateurs responsables sont seuls habilités à mettre en place et à administrer les systèmes d'identification et d'authentification des utilisateurs, conformes aux directives du comité de coordination SSI. Il en va de même pour les dispositifs de contrôle d'accès aux systèmes, aux réseaux et aux données.

Sauf exception formulée par un document écrit signé d'un responsable d'entité, seuls l'administrateur local et ses collaborateurs immédiats possèdent les droits d'administrateur sur les postes de travail des utilisateurs des SI de l'INSIGU.

Audits périodiques

Les administrateurs procèdent deux fois par an à un audit des comptes des utilisateurs et des droits d'accès associés, pour vérifier leur validité et leur exactitude.

Mise en œuvre et litiges

Rapport des violations des règles SSI

Pour toute violation des règles SSI qu'il est amené à constater, l'administrateur établit un rapport écrit destiné au comité de coordination SSI et à ses responsables hiérarchiques.

Veille SSI

Les administrateurs exercent régulièrement une activité de veille scientifique et technologique dans le domaine SSI. Ils sont abonnés aux listes de diffusion qui publient les découvertes de vulnérabilités. Ils participent notamment aux activités de formation, d'information et de sensibilisation entreprises par le comité de coordination SSI.

Attitude à l'égard des violations de la loi

Lorsque l'administrateur constate des violations de la loi dans son périmètre d'activité, il en fait rapport au comité de coordination SSI et à ses responsables hiérarchiques, qui prendront les mesures adéquates afin de coordonner leurs actions avec les autorités judiciaires.

Attitude à l'égard des violations des règles SSI

La direction de l'INSIGU, ou son représentant qualifié, peut révoquer le compte et les droits d'accès au réseau et aux données d'un utilisateur qui aurait violé les règles SSI mentionnées dans la Charte de l'utilisateur des ressources informatiques et des services Internet de l'INSIGU.

Quatrième partie

Avenir de la sécurité du système d'information

10

Nouveaux protocoles, nouvelles menaces

Depuis quelques années, des protocoles qui ne sont peut-être plus nouveaux d'un point de vue chronologique, mais qui méritent encore ce qualificatif par l'innovation qu'ils ont incarnée par rapport aux protocoles traditionnels de l'Internet, fondés sur le modèle client-serveur, posent aux administrateurs de réseaux de nouvelles questions, notamment dans le domaine de la sécurité. Le présent chapitre, après un rappel du modèle traditionnel, présente les deux principales familles de ces protocoles novateurs : les systèmes poste à poste *(peer to peer)* et la téléphonie par Internet.

Le modèle client-serveur

Dans le modèle traditionnel, un utilisateur de l'Internet agit au moyen d'un ordinateur équipé d'un logiciel appelé *client* qui s'adresse, à distance, à un logiciel *serveur*.

Ainsi, le logiciel avec lequel vous écrivez votre courrier électronique est un client de messagerie, ou selon le jargon technique un *Mail User Agent (MUA, UA)*, ou encore, pour le désigner par le protocole employé pour expédier le courrier, un *client SMTP (Simple Mail Transport Protocol)*. Ce client va établir une communication avec un *serveur SMTP*, encore appelé *Mail Transfer Agent (MTA)* ou passerelle de messagerie, avec lequel il va d'abord échanger quelques données de service afin que l'un et l'autre identifient leur interlocuteur et la nature des échanges à venir, puis le client va envoyer au serveur des messages que celui-ci se chargera de faire parvenir à leurs destinataires, éventuellement par l'intermédiaire d'autres MTA, dits *relais*. Notons que votre logiciel de courrier, qui est un client SMTP pour envoyer des messages, est pour les recevoir un client *Post Office Protocol (POP)* ou *Internet Message Access Protocol (IMAP)*, en effet, en règle générale, ce ne sont pas les mêmes protocoles qui servent à émettre et à recevoir des messages de courrier électronique.

De la même façon, le logiciel navigateur avec lequel vous explorez le Web, que ce soit *Internet Explorer*, *Safari* ou *Firefox*, est un *client Web* qui s'adresse à un *serveur Web* (souvent animé par le logiciel *Apache*) pour lui demander de lui envoyer les pages que vous désirez consulter. Les communications auront lieu selon le protocole *HyperText Transport Protocol (HTTP)*.

À chacun des protocoles que nous avons évoqués est attribué, par convention, un numéro de port[1], et le serveur du protocole écoute les connexions entrantes en provenance du réseau qui comportent ce numéro comme port de destination ; c'est ainsi que les serveurs détectent les connexions qui leur sont destinées : port 25 pour SMTP, 80 pour HTTP, 110 pour POP3, 143 pour IMAP, 137, 138, 139 et 445 pour Netbios et les services associés, etc.

Au bon vieux temps où les protocoles fonctionnaient ainsi, la sécurité du réseau était un jeu d'enfant (enfin presque). Un simple routeur muni de listes de contrôles d'accès (ACL) pouvait faire office de pare-feu : si le réseau comporte un serveur Web public, j'autorise les connexions entrantes sur le port 80 à destination de son adresse IP, et je les interdis pour toutes les autres adresses. Si j'ai une passerelle de messagerie (MTA), j'autorise le trafic SMTP sortant à partir de son adresse, et uniquement à partir de celle-là, notamment parce que beaucoup de virus modernes comportent un petit agent SMTP pour envoyer des informations

[1]Pour la définition du *port*, voir l'encadré page 121.

à leur maître (ou du courriel non sollicité à des millions d'internautes !) Je n'autorise *a priori* aucune connexion entrante, le seul trafic entrant sera constitué de connexions initialisées à partir de l'intérieur du réseau, sauf pour les serveurs publics dûment répertoriés et placés en DMZ (voir le chapitre 6 page 105). Cela s'appelle le filtrage par port, et les gens qui faisaient cela soigneusement étaient jusqu'à ces dernières années relativement à l'abri des mauvaises surprises, ils pouvaient se dire que leur réseau était raisonnablement bien protégé.

Il faut continuer à faire soigneusement du filtrage par port, mais cela ne suffit plus : le monde a changé pour devenir plus cruel !

Versatilité des protocoles : encapsulation HTTP

Tous en HTTP !

Le premier coup de hache dans le modèle du filtrage par port est venu de l'universalité du protocole HTTP sur le port 80 : comme à peu près tous les réseaux comportent un serveur Web et laissent de ce fait circuler librement les connexions à destination du port 80, des développeurs de protocoles astucieux encapsulent leurs paquets de données dans des paquets HTTP, ce qui leur permet de franchir les pare-feu sans encombre avant d'être « décapsulés » pour accomplir leur mission. Il est aussi assez courant de recourir au même procédé avec la version chiffrée du protocole, HTTPS (port 443), ce qui ajoute une difficulté : les paquets encapsulés sont chiffrés et il est donc impossible de les analyser, même pour un pare-feu qui ferait de l'« inspection en profondeur ». Un pare-feu qui se fie aux numéros de port n'y voit... que du feu !

HTTPS n'est rien d'autre que HTTP encapsulé dans TLS*(Transport Layer Security)*. En général le serveur est authentifié par un certificat X509, l'internaute peut s'authentifier par l'intermédiaire d'un serveur RADIUS (c'est une des meilleures méthodes), ou par un des autres procédés proposés par les logiciels serveur.

Vertus de HTTPS

L'encapsulation de tout et de n'importe quoi dans un protocole omniprésent comme HTTP/HTTPS crée des difficultés au responsable de sécurité, mais peut aussi lui procurer des solutions à quelques problèmes.

En fait HTTPS a permis un regain d'essor de l'Internet, en facilitant considérablement la mise en place de plateformes de commerce électronique : au départ technologie de pointe réservée à de grandes institutions financières, le paiement en ligne est aujourd'hui accessible aux PME pour un prix abordable et dans de bonnes conditions de sécurité.

HTTPS est également un candidat prometteur pour le remplacement des applications client-serveur : la substitution est séduisante, parce que, avec les solutions client-serveur traditionnelles, le logiciel client doit être déployé sur tous les postes de travail, alors que le navigateur nécessaire à HTTPS est déjà déployé partout. Des langages comme PHP et JavaScript ont rendu le développement facile ; si vraiment le projet est très volumineux ou complexe, on peut utiliser Java.

Protocoles poste à poste (peer to peer)

Définition et usage du poste à poste

Le second coup de hache est venu des protocoles *peer to peer* (souvent abrégé en P2P), ce que Wikipédia propose de traduire en français par *poste à poste* et décrit ainsi :

« P2P désigne un modèle de réseau informatique dont les éléments (les nœuds) ne jouent pas exclusivement le rôle de client ou de serveur mais fonctionnent des deux façons, en étant à la fois clients et serveurs des autres nœuds de ces réseaux, contrairement aux systèmes de type client-serveur, au sens habituel du terme. (...)

Les réseaux P2P permettent de communiquer et de partager facilement de l'information — des fichiers le plus souvent, mais également des calculs, du contenu multimédia en continu *(streaming)*, etc. sur Internet. Les technologies P2P se sont d'ailleurs montrées si efficaces que le P2P est considéré par certains comme l'étape ultime « de la liberté et de la démocratie » sur Internet. Sans aller jusque-là, on considère souvent que le P2P porte (et est porté par) une philosophie de partage et un profond esprit communautaire. »

Pour une présentation des évolutions récentes on pourra consulter la communication de Franck Cappello aux journées JRES 2005 [26].

Ces protocoles poste à poste sont utilisés massivement par les internautes équipés d'une connexion à haut débit pour échanger des fichiers aux contenus musicaux ou cinématographiques, au titre de ce que le droit français nomme la *copie privée*, et le droit américain *fair use*.

Les industries du disque et du cinéma n'étaient pas préparées à cette extension de la copie privée, à laquelle elles ont réagi principalement par le recours à la loi. Les premiers protocoles P2P, tel Napster, comportaient un serveur central qui recueillait et distribuait les adresses des participants, ce qui a permis aux industriels d'engager contre le propriétaire de ce serveur des actions en justice, et d'obtenir sa fermeture en 2001. Napster est devenu maintenant un site de téléchargement légal de musique, en accord avec les ayant-droit.

Après cette expérience, les protocoles poste à poste actuels, tels KaZaA, Skype, eMule ou BitTorrent, ne comportent plus de serveur central, ce qui oblige les entreprises qui souhaiteraient poursuivre leurs utilisateurs à les identifier un par un.

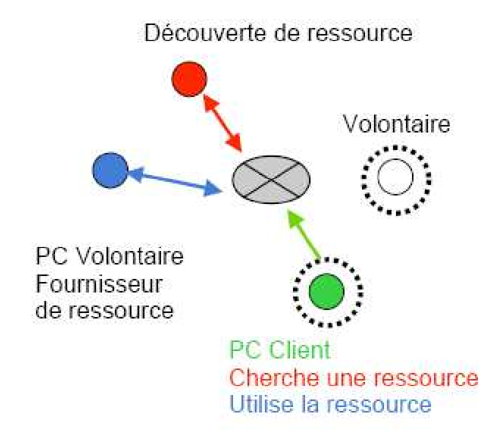
Problèmes à résoudre par le poste à poste

Les nœuds des systèmes poste à poste, quasiment par définition, sont des ordinateurs situés à la périphérie de l'Internet, et qui sont le plus souvent soit des machines personnelles dans un domicile privé, soit des postes de travail individuels au sein d'une entreprise qui n'a pas vraiment prévu qu'ils soient utilisés pour du poste à poste, voire qui essaye de l'interdire. Les conséquences techniques de cette situation sont les suivantes :

- les ordinateurs concernés sont souvent éteints ;
- ils n'ont souvent pas d'adresse IP permanente...
- ... voire pas d'adresse routable (adresses dites « NAT *(Network Address Translation)* », cf. page 147).

Il faudra, malgré ce contexte d'amateurisme, que tous les nœuds puissent être à la fois clients et serveurs, qu'ils puissent communiquer directement deux à deux, et que chacun en fonction de ses capacités contribue au fonctionnement général de l'infrastructure. Il faut qu'un nœud qui rejoint le réseau puisse *découvrir* ceux qui offrent les ressources qui l'intéressent, selon le schéma de la figure 10.1 page suivante.

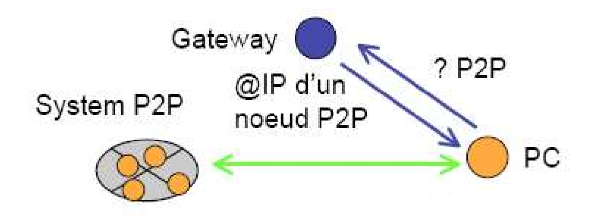
Figure 10.1
Un poste client tente de rejoindre une communauté de pairs.



Pour surmonter les difficultés énumérées plus haut et atteindre ces objectifs, un système poste à poste comporte quatre composants fondamentaux :

1. une passerelle, qui publie l'adresse IP d'autres nœuds et permet à l'utilisateur de choisir une communauté au sein de laquelle il va échanger des données, comme représenté par la figure 10.2 ;

Figure 10.2
Une passerelle *(gateway)* va permettre au nouvel arrivant de découvrir l'adresse IP d'un membre déjà connecté.



2. un protocole réseau pour l'établissement des connexions et l'exécution des opérations de transport de données ; un élément crucial de ce protocole sera bien sûr son aptitude au franchissement de pare-feu, comme indiqué par la figure 10.3 ; en effet la communication poste à poste serait impossible dans le respect des règles de filtrage qu'imposent la plupart des réseaux, notamment en entreprise ;

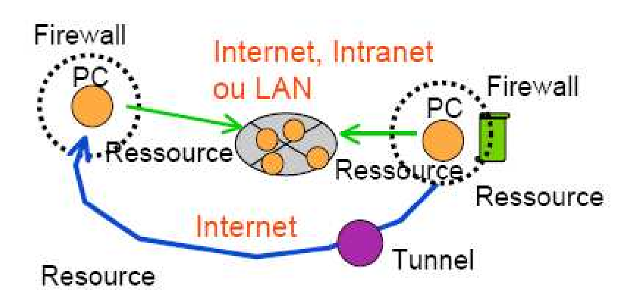


Figure 10.3
Ici deux nœuds confinés par des pare-feu *(firewalls)* essaient néanmoins de construire une voie de communication entre eux, mais le procédé retenu est rudimentaire et peu efficace.

3. un système de publication de services et d'annonces de ressources, qui permet à chacun de contribuer à l'œuvre commune ;
4. un système, symétrique du précédent, de recherche de ressources, qui permet de trouver ce que l'on cherche, tel morceau de musique ou tel film, ou le chemin d'accès à tel téléphone réseau.

Le poste à poste et la sécurité

Filtrer tel ou tel protocole poste à poste est un objectif que peut se fixer un administrateur de réseau, ou qui peut lui être fixé par son employeur ou son client. En effet certains de ces protocoles peuvent être utilisés à des fins qui enfreignent les législations relatives à la propriété industrielle, et ils peuvent aussi contrevenir aux règles de sécurité d'un organisme.

Dans l'Internet traditionnel, un tel objectif pouvait être atteint par le procédé du filtrage de port, que nous avons mentionné aux pages 123 et 207. Avec les protocoles poste à poste, mais aussi avec d'autres protocoles, comme H323, destiné à

acheminer la voix et la vidéo sur IP, le filtrage par port est impossible parce qu'ils font des ports un usage dynamique, c'est-à-dire qu'ils utilisent des numéros de port variables et qu'éventuellement ils en changent en cours de session.

Exemples : KaZaA et Skype

KaZaA est un système P2P d'échange de fichiers, aujourd'hui un peu démodé. Skype, lancé par la même équipe, est un système de téléphonie par Internet. Les deux ont eu un succès considérable, basé sur un même modèle : pour la plupart des usages, c'est gratuit, sans être d'ailleurs libre, puisque le code source du logiciel n'est pas public. Les protocoles sont secrets, et surtout furtifs. Ils fonctionnent en l'absence de tout serveur central (enfin presque, en ce qui concerne Skype), ce qui leur évite les désagréments subis en son temps par le créateur de Napster. Ainsi, même si certains internautes en font un usage contraire aux lois, les auteurs ne peuvent être poursuivis.

Il semble bien que la plus grande part des échanges de données effectués au moyen de KaZaA concerne des enregistrements musicaux et cinématographiques, et que ces échanges ne soient pas approuvés par les titulaires des droits d'auteur des œuvres en question. Les entreprises soucieuses d'éviter les embarras juridiques seront bien inspirées d'interdire à leurs employés d'utiliser KaZaA sur leur lieu de travail, d'autant plus que les ordinateurs des employés qui l'utiliseraient peuvent, à leur insu, devenir des *serveurs* KaZaA et diffuser musique et films à la planète entière, ce qui en général attire l'attention des ayants droit et peut occasionner de lourdes condamnations.

Le succès public de ces deux systèmes a été considérable. Nous examinerons plus en détail Skype, le plus récent.

Description de Skype

Skype est un système poste à poste de communication vocale sur IP lancé en août 2003 par la société luxembourgeoise Skype Technologies S.A., fondée par Janus Friis et Niklas Zennstrom, les créateurs de KaZaA; cette société a été rachetée par eBay en septembre 2005. Comme pour KaZaA, les clients Skype cherchent sur le réseau d'autres clients Skype avec lesquels ils vont entrer en communication, et à partir de là ils rejoignent un réseau virtuel au moyen duquel ils vont tenter de localiser les correspondants avec lesquels ils souhaitent établir une communica-

tion. À la différence de KaZaA, qui était financé essentiellement par la publicité, le système Skype tire ses revenus de la facturation d'un service payant, celui des passerelles qui permettent à ses utilisateurs d'entrer en communication avec un abonné du réseau téléphonique ordinaire ; les méthodes d'accès à ces passerelles s'appellent *SkypeIn* et *SkypeOut*.

Skype est disponible pour les systèmes Windows, MacOS X, PocketPC et Linux. Il permet l'établissement d'une communication gratuite et directe entre deux ordinateurs équipés du logiciel Skype, d'un micro et d'un haut-parleur, via l'Internet. Skype comporte également un système de messagerie instantanée qui permet l'échange de messages écrits et de fichiers. Les passerelles payantes permettent d'atteindre un abonné au téléphone ordinaire.

Skype permet également l'organisation de téléconférences et, depuis la version 2.0 de janvier 2006, de visioconférences, à condition que les participants soient équipés de webcams. De l'avis des utilisateurs, il semble que la qualité du son soit excellente, meilleure que celle des concurrents. Les communications qui circulent sur le réseau sont chiffrées, ce qui assure une bonne confidentialité.

Comme le souligne Simson L. Garfinkel [58], le succès de Skype est dû notamment au fait qu'il est bien plus facile à installer et à utiliser que les systèmes concurrents. De surcroît, Skype, comme KaZaA, est conçu de façon à pouvoir franchir sans encombre les pare-feu et les dispositifs de traduction d'adresse (NAT). L'efficacité des méthodes de franchissement de Skype n'est pas sans laisser craindre à certains employeurs qu'elles soient utilisées par leurs employés pour des usages privés, et les administrateurs de ces réseaux, ainsi que les responsables de sécurité, ne laissent pas d'être agacés par ce logiciel conçu spécialement pour faire échec aux mesures de sécurité et de filtrage qu'ils s'efforcent de mettre en place. On craint également que des nœuds espions puissent être introduits dans le réseau Skype.

Le protocole utilisé par Skype n'est pas publié, non plus que le code source du logiciel. Néanmoins ce dernier a pu être analysé par Fabrice Desclaux [39] et Philippe Biondi [16]. La tâche n'est pas facile : le programme exécutable est chiffré sur le disque dur, et n'est déchiffré qu'une fois chargé en mémoire vive. En plusieurs dizaines d'emplacements le programme vérifie l'intégrité de son propre texte, de façon à éviter qu'un candidat à la rétro-ingénierie du protocole ou du code ne réussisse à l'instrumenter.

Skype est-il vraiment poste à poste ?

Une fois la communication établie entre deux postes, le fonctionnement de Skype obéit bien aux principes « poste à poste ». Néanmoins plusieurs aspects du système s'écartent de ces principes.

L'utilisateur de Skype s'identifie auprès du serveur central `ui.skype.com`, situé semble-t-il aux Pays-Bas, qui effectue l'authentification des utilisateurs et des logiciels Skype. L'authentification repose sur une signature par biclé RSA.

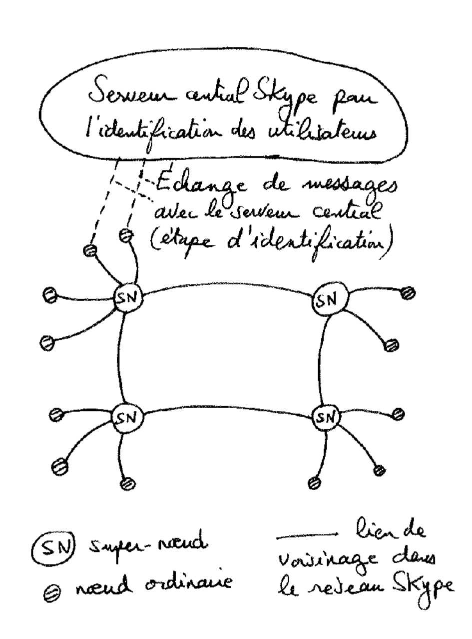


Figure 10.4
Principe de fonctionnement de Skype (source : Salman A. Baset et Henning Schulzrinne).

Le réseau Skype comporte des « supernœuds » ; tout nœud Skype peut devenir supernœud, à l'insu de son utilisateur, s'il dispose d'une adresse IP publique et qu'il n'est pas situé derrière un pare-feu. Les supernœuds peuvent ainsi servir de relais et de mandataires aux nœuds moins bien lotis, qui les découvrent par des procédés non publiés. Skype n'offre aucun dispositif qui permettrait à un utilisateur d'interdire à sa station de devenir supernœud.

Les méthodes SkypeIn et SkypeOut pour téléphoner à des abonnés de réseaux ordinaires reposent sur des serveurs Skype répartis dans différents pays.

Sécurité de Skype et d'autres protocoles

Simson L. Garfinkel [58], Salman A. Baset et Henning Schulzrinne [14] se sont livrés à une analyse détaillée du système, et notamment des protocoles et des dispositifs de sécurité qu'il utilise. Garfinkel signale un certain nombre de moyens qui pourraient permettre à un attaquant de compromettre le système sur le poste de travail d'un utilisateur, de retrouver l'historique des messages instantanés échangés, de compromettre des couples nom d'utilisateur – mot de passe. Ces failles sont plutôt moins nombreuses et moins graves que celles d'autres systèmes analogues dont le fonctionnement suppose souvent l'abolition de toute mesure de sécurité sérieuse : nous pensons à l'utilisation naïve de systèmes de visioconférence basés sur le protocole H323, dont l'utilisation à peu près sûre requiert les précautions suivantes :

1. mise en place d'un mandataire, par exemple *Gatekeeper* ;
2. utilisation de postes de travail dépourvus de toutes données sensibles, sur un sous-réseau réservé à cet usage.

En fait, la nature même des usages de ces protocoles les expose à des compromissions, qui sont du même ordre que celles qui affectent le réseau téléphonique ordinaire, même si les techniques d'attaque sont différentes : les délais d'établissement et d'exécution de la communication la rendent difficilement furtive. Les précautions à prendre peuvent difficilement recourir à un chiffrement supplémentaire sur le poste client. À l'intérieur d'une entreprise, l'usage au sein d'un VPN correctement configuré peut être une solution.

Filtrer Skype ?

Filtrer les protocoles de Skype au moyen d'un pare-feu classique s'avère une entreprise compliquée. En effet Skype est conçu pour éviter ce filtrage : les numéros de port sont variables, les transactions mettent en jeu différentes stations. Néanmoins, il existe des moyens de filtrer assez efficacement Skype ou KaZaA (le problème à résoudre est similaire). Les méthodes employées sont heuristiques et reposent sur l'analyse comportementale du protocole. Il faut observer pendant un certain laps de temps le déroulement des échanges (ce qui suppose leur journalisation) ; l'analyse de ces observations permet la détection d'une « signature »

des transactions Skype, notamment par l'identification de séquences et de temporisations caractéristiques. Une implémentation d'une telle méthode existe sur la base du logiciel libre *IP Tables / Netfilter*, qui permet de réaliser un pare-feu (cf. page 125).

Nous reviendrons sur ces questions du filtrage des protocoles poste-à-poste au chapitre suivant, à la page 238, lorsque nous verrons les mesures de prohibition et de rétorsion contre les échanges poste à poste envisagés par les éditeurs de films et de disques.

Franchir les pare-feu : vers une norme ?

Comme nous l'avons noté ci-dessus, le franchissement des pare-feu et des dispositifs de traduction d'adresses est un des problèmes à résoudre pour faire du poste à poste au grand large. Pour remédier à la situation présente, où chaque protocole utilise sa propre méthode plus ou moins efficace et plus ou moins heuristique, est apparu à l'IETF *Internet Engineering Task Force* le projet de RFC 3489 : STUN *(Simple Traversal of UDP Trough NAT)*. L'idée est la suivante : comme de toute façon les candidats au franchissement arriveront à leurs fins, autant leur permettre de le faire avec un minimum de conformité avec l'orthodoxie protocolaire. Yves Drothier, du *Journal du Net*, décrit la chose ainsi :

« L'intérêt de STUN est de reconnaître les dispositifs de sécurité placés entre le routeur NAT (les routeurs NAT agissent comme des pare-feu, faisant le lien entre les adresses privées et les adresses publiques lors de communication IP) et le réseau public afin d'établir les communications malgré le filtrage.

STUN identifie les différents dispositifs de sécurité NAT en émettant un message de l'infrastructure cliente vers le serveur STUN situé en aval du routeur NAT. Ce message explore ainsi quels sont les ports et les adresses IP utilisés par les dispositifs de sécurité NAT pour router le message. Ce sont ces données qui seront utilisées par la suite lors d'appels entrants ou sortants pour établir la communication. »

Comme STUN ne résout pas tous les problèmes, un autre protocole, *Traversal Using Relay NAT (TURN)*, a été mis au point notamment pour les configurations avec traduction d'adresse symétrique *(symmetric NAT)*. Un routeur NAT symétrique établit un chemin en fonction de l'adresse IP de l'émetteur et de son port d'accès mais aussi en fonction de ces mêmes informations chez le destinataire. Le

chemin ainsi créé échappe au serveur STUN puisqu'il peut changer en fonction du destinataire appelé.

Téléphonie IP : quelques remarques

La téléphonie sur IP *(Internet Protocol)* et sa sécurité constituent un sujet qui mériterait un livre entier, nous n'aurons donc pas la prétention de le traiter ici, mais nous livrons au lecteur quelques remarques de mise en garde qui doivent beaucoup à des exposés d'Hervé Schauer[2], de Nicolas Fischbach[3] et de Loïc Pasquiet[4] auxquels le lecteur pourra se reporter pour de plus amples développements.

La première chose à signaler au sujet de la téléphonie sur IP, c'est qu'elle est aujourd'hui inévitable : si l'on consulte les fournisseurs de matériel téléphonique pour un projet qui dépasse la dizaine de postes, toutes les réponses seront en téléphonie sur IP, ou peu s'en faut. Il est donc pratiquement impossible d'acheter autre chose, la téléphonie traditionnelle a vécu, bientôt elle ne sera plus maintenue.

Une grande variété de protocoles peu sûrs

La transmission de la voix sur IP recouvre en fait une grande variété de protocoles :

- H323 est un protocole (en voie de disparition pour la téléphonie) adapté d'ISDN et normalisé par l'Union internationale des télécommunications. Ce protocole, par sa conception, n'offre pas de bonnes garanties de sécurité : les risques encourus sont l'écoute des communications, l'usurpation d'identité et le déni de service.
- SIP est un protocole de signalisation normalisé par l'IETF (RFC 3261) ; il doit être associé pour le transport de la voix à un autre protocole, qui peut être RTP, RTCP ou RTSP. Les risques encourus sont les mêmes qu'avec H323.
- SCCP est un protocole de la maison *Cisco* ; les risques sont toujours les mêmes, ainsi d'ailleurs que pour les protocoles privés des autres fournisseurs (Alcatel, Avaya...).

[2] http://www.hsc.fr/ressources/presentations/tenor06-voip-sec/
[3] http://www.ossir.org/jssi/jssi2006/supports/1B.pdf
[4] http://www.ossir.org/jssi/jssi2006/supports/2A.pdf

- MGCP *(Multimedia Gateway Control Protocol)* est normalisé par l'IETF (RFC 3435) et offre de meilleures garanties de sécurité que les précédents. Ce protocole est déployé par les opérateurs sur leurs réseaux ADSL.
- Signalons aussi plusieurs méthodes d'encapsulation de GSM dans IP, par lesquelles les opérateurs tentent de lutter contre Skype.

Ainsi, aucun de ces protocoles n'offre intrinsèquement de bonnes garanties de sécurité, même si Hervé Schauer crédite MGCP d'une plus grande sûreté due notamment à l'absence de toute fonction « intelligente » dans le poste téléphonique. Ce qui ne veut pas dire que l'on ne peut pas les utiliser en prenant des précautions supplémentaires.

Précautions pour la téléphonie IP

Les précautions à prendre pour la téléphonie IP sont somme toute classiques :

- cloisonnement des réseaux, spécialisation des VLAN ;
- filtrage des adresses MAC et du trafic IP ;
- authentification et chiffrement, etc.

À quoi s'ajoute la mise en service des fonctions de sécurité sur les terminaux téléphoniques. Ces fonctions de sécurité sont notamment destinées à empêcher l'usurpation d'identité et l'écoute des communications. Il est à noter que pour disposer de ces fonctions indispensables il faut renouveler entièrement le parc de terminaux téléphoniques, et ne pas se contenter des matériels « premier prix ».

Il faut avoir en tête les points suivants :

1. Le réseau téléphonique classique est un élément crucial de la sécurité des personnes et des biens : il permet de donner l'alerte en cas d'accident de personne, d'incendie, d'acte criminel ; il doit donc fonctionner sept jours sur sept, 24 heures sur 24. Adopter la téléphonie sur IP impose que le réseau informatique soit soumis aux mêmes exigences, or actuellement ce n'est pas le cas. Il faudra donc augmenter la capacité des équipements actifs, puis les doubler, et prévoir du personnel en astreinte la nuit et les jours fériés. Sans oublier les installations destinées à secourir l'alimentation électrique de ces équipements. Les conséquences financières et organisationnelles ne sont pas négligeables.
2. Des services informatiques comme le DNS ou DHCP deviennent critiques.

3. Dans de nombreux cas, les entreprises qui ont adopté la téléphonie sur IP ont été amenées à renouveler entièrement ou en grande partie leur parc de terminaux, en sachant que les terminaux les moins chers ne donnent généralement pas satisfaction, notamment parce qu'ils sont dépourvus des fonctions de sécurité.
4. La perspective de faire des économies grâce au partage du câblage est le plus souvent illusoire : les systèmes d'authentification de type 802.1x exigent une prise par équipement, le service qui gère les terminaux a également besoin d'identifier les prises sur lesquelles ils sont branchés, la commodité d'usage des terminaux impose souvent une alimentation électrique par la prise réseau ; tout cela conduit en général à l'installation d'un câblage particulier pour la téléphonie sur IP.
5. Faire fonctionner un système de téléphonie IP demande des compétences en réseaux informatiques : le personnel des services généraux en est dépourvu, et sera incapable de mettre en œuvre les fonctions de sécurité. L'acquisition de compétences élémentaires en réseau est possible, par exemple en cours du soir au CNAM, en deux ans, en partant d'un niveau baccalauréat scientifique.

11

Tendances des pratiques de sécurisation des SI

La fin de l'été 2005 a vu la publication de deux articles (*The Six Dumbest Ideas in Computer Security*[1] de Marcus J. Ranum et *The Next 50 Years of Computer Security : An Interview with Alan Cox* par Edd Dumbill[2]) qui sont appelés à faire date dans le domaine de la sécurité informatique : en effet ils réservent un sort cruel à quelques idées reçues et à quelques intuitions largement partagées. Le présent chapitre, à la faveur d'une étude de ces articles, aborde plusieurs questions fondamentales : les systèmes de détection ou de prévention d'intrusion, ainsi que la protection de la propriété *et* de la liberté intellectuelles dans un monde numérique.

---

[1]Cf. `http://www.ranum.com/security/computer_security/editorials/dumb/`
[2]Cf. `http://www.oreillynet.com/pub/a/network/2005/09/12/alan-cox.html`

Les six idées les plus stupides en sécurité, selon Ranum

Marcus J. Ranum m'a autorisé à faire ici de larges emprunts à son article, qu'il en soit remercié. S'il fallait résumer en une idée générale les thèses qu'il défend et qu'il illustre, ce serait que, si l'on veut construire un système informatique (au sens large) sûr, il faut que la sécurité soit incorporée à sa conception dès l'origine : il est coûteux et inefficace de vouloir « ajouter de la sécurité » *a posteriori* à un système conçu sans idée de sécurité au départ. Le corollaire de cette idée, c'est qu'il est possible de concevoir un tel système, que les méthodes existent pour ce faire, et M.J. Ranum en donne quelques exemples. Nous avons d'ailleurs eu l'occasion à la page 89 de décrire un système conçu selon ces principes dès 1964, Multics.

Marcus J. Ranum est un pionnier de la sécurité des systèmes d'information ; inventeur de la notion de pare-feu *(firewall)*, il en a également signé la première réalisation à la fin des années 1980 ; il a aussi joué un rôle précurseur dans le développement des systèmes de détection d'intrusion. Nous allons présenter et discuter ses six propositions[3]. Mais nous pouvons, avant de commencer, être déjà d'accord avec lui pour dire que si votre politique de sécurité est indigente et si les règles que vous fixez sont insuffisantes ou incohérentes, aucun pare-feu de grand luxe ne protégera votre site, eût-il coûté 100 000 euros.

Idée stupide nº 1 : par défaut, tout est autorisé

Cette idée ne demande pas un long examen pour être classée en première place dans la liste des stupidités. Il est assez clair que les conditions actuelles sur les réseaux exigent que par défaut tout soit interdit, et que ne soient autorisées que les actions effectivement et positivement identifiées comme légitimes. Mais cette idée stupide, si facile à réfuter en apparence, est incroyablement résiliente et envahissante.

C'est bien sûr dans la rédaction des règles de pare-feu que cette idée stupide numéro 1 se manifeste en priorité : on laisse passer par défaut tous les types de trafic et on bloque ceux que l'on estime dangereux ; une variante consiste à bloquer pas mal de choses mais à aligner une longue liste de dérogations qui, outre le fait qu'elles vont détériorer les performances de l'accès au réseau, vont anéantir la sécurité, parce que ces dérogations seront autant de portes assez faciles à ouvrir, par exemple par usurpation d'adresse IP, l'enfance de l'art pour un pirate amateur.

[3] `http://www.ranum.com/security/computer_security/editorials/dumb/`

Nous serons d'accord avec M. Ranum pour dire que la véritable bonne idée, c'est « par défaut, tout est interdit ».

Idée stupide nº 2 : prétendre dresser la liste des menaces

Cette idée stupide numéro 2, en fait assez voisine de sa sœur la numéro 1, pourrait aussi s'incarner dans une configuration de pare-feu établie en fonction de la liste des menaces recensées. Elle est stupide car en 2006 la liste des menaces est très longue, et surtout elle s'accroît chaque jour : les recenser pour mettre son pare-feu à jour s'apparente au remplissage du tonneau des Danaïdes. Les listes auxquelles je suis abonné publient une dizaine de nouvelles *vulnérabilités* par semaine, et on estime entre 200 et 700 par mois le nombre de nouvelles *menaces*.

Le délai qui s'écoule entre la découverte d'une vulnérabilité et son exploitation par un logiciel menaçant est passé en quelques années de quelques mois à une quinzaine de jours dans certains cas. C'est-à-dire que le logiciel nuisible peut apparaître avant la correction de la vulnérabilité, et que même si ce n'est pas le cas il peut suffire d'un retard de quelques heures dans l'application de la correction pour être exposé sans défense à la menace. Et n'oublions pas que les pirates, présents dans tous les fuseaux horaires, agissent durant nos nuits et nos jours fériés. Bref, en 2006 il est effectivement stupide d'espérer assurer la sécurité de son SI en se prémunissant contre des menaces qui seraient connues d'avance.

Il faut au contraire dresser la liste de tous les logiciels *utiles*, d'usage légitime dans le SI de l'entreprise, et interdire tous les autres en vertu de la règle précédente.

Ainsi, considérons un projet de sécurité informatique destiné à évaluer et à améliorer la disponibilité d'un système d'information. Si le responsable du projet s'inspire de la méthode EBIOS élaborée en France par la Direction centrale de la sécurité des systèmes d'information (DCSSI), il dressera une liste des risques, associera chacun de ces risques à des vulnérabilités, et envisagera les contre-mesures qu'il peut élaborer pour s'en prémunir, selon une formule pleine de bon sens et d'utilité[4] :

$$\text{risque} = \frac{\text{menace} \times \text{vulnérabilité} \times \text{sensibilité}}{\text{contre-mesure}}$$

[4] Nous avons donné à la page 7 une autre formule pour le risque, qui complète utilement celle qui va suivre.

Cette conceptualisation paraît intéressante, la formule multiplicative permet de classer les risques selon un ordre de priorité en fonction de leur intensité concrète pour l'entreprise, par opposition à une intensité technique perçue par l'ingénieur de sécurité, mais elle peut engendrer la tentation de dresser une liste de risques ou une liste de vulnérabilités que l'on placera dans la colonne de gauche d'un tableau, afin d'en remplir la colonne de droite avec les contre-mesures appropriées.

Pourquoi cette démarche est-elle maladroite ? Parce que les risques et les menaces sont nombreux et souvent inconnus, alors que le répertoire des contre-mesures possibles est beaucoup plus réduit ; souvent, cela peut se résumer à cinq ou six grands thèmes : plan de sauvegarde des données, amélioration du stockage, aménagement d'un site de secours avec duplication des données à distance, administration correcte des serveurs (fermeture des services inutiles, séparation des privilèges, application des correctifs de sécurité, surveillance des journaux), sécurisation du réseau (pare-feu, authentification forte, fermeture des services inutiles), sécurisation physique des locaux. Il est donc plus simple et plus efficace de partir de la table inverse de la précédente : mettre les contre-mesures dans la colonne de gauche, et énumérer dans la colonne de droite les risques éliminés par elles, ce qui évitera de payer un consultant pendant des mois pour élaborer la liste des centaines de risques plus ou moins réels que l'on peut envisager.

Idée stupide n° 3 : tester par intrusion, puis corriger

La mise en pratique de cette idée stupide numéro 3 consiste à détecter les failles du système à protéger en perpétrant une intrusion, en d'autres termes, à attaquer son système de protection, pare-feu, antivirus ou autre, puis à obturer les brèches que l'on aura détectées. Cette idée stupide est mise en œuvre par de nombreux cabinets spécialisés, qui proposent des *tests d'intrusion* à leurs clients, lesquels, lorsqu'ils sont incompétents, sont friands de ce genre d'exercice.

M. Ranum observe que, si la sécurité par test d'intrusion et correction était une bonne méthode, les failles d'*Internet Explorer* seraient corrigées depuis longtemps. Il observe également que certains logiciels, comme *Postfix* ou *Qmail*, sont quasiment exempts de failles depuis leur naissance, et ce parce qu'ils ont été conçus dès l'origine pour ne pas en comporter, c'est-à-dire que leur réalisation s'est appuyée sur des méthodes à l'épreuve des failles.

M. Ranum en vient là à son idée centrale : la seule façon d'obtenir un système sûr, c'est qu'il le soit dès la conception, et c'est possible. Nous pourrions qualifier ce principe de *méthode de sécurité a priori*, par opposition aux méthodes de sécurité *a posteriori*, qui consistent à construire des systèmes non sûrs, puis à essayer de les réparer en détectant les failles *a posteriori*. Par analogie, nous pourrions dire que la méthode en usage dans la Marine Nationale et connue par la devise « Peinture sur rouille[5] égale propreté » ne donne pas en matière de sécurité des résultats satisfaisants.

M. Ranum conclut sur ce point en indiquant que si votre système est régulièrement vulnérable au « bug de la semaine », c'est que vous êtes dans la configuration évoquée ici, et que tout pirate qui inventera une attaque nouvelle réussira chez vous.

Idée stupide n° 4 : les pirates sont sympas

« La meilleure façon de se débarrasser des cafards dans la cuisine, c'est de jeter les miettes de pain sous la cuisinière, c'est bien connu », nous dit ironiquement M. Ranum, avant de citer Donn Parker :

« L'informatique en réseau a affranchi les criminels de la contrainte historique de proximité avec leur crime. L'anonymat et l'exemption de la confrontation personnelle avec la victime ont diminué la difficulté émotionnelle à commettre un crime, parce que la victime n'est qu'un ordinateur inanimé, pas une personne ou une entreprise réelles. Les gens timides peuvent se mettre au crime. La prolifération de systèmes identiques, de moyens d'y accéder et l'automatisation des transactions commerciales permettent et favorisent l'économie du crime automatisé, la réalisation d'outils criminels de grande puissance et l'apparition de scénarios très rentables. »

La criminalité informatique est un problème social, pas une question de technologie, nous dit M.J. Ranum. La diffusion de l'informatique a donné un champ d'action élargi à certaines personnes dépourvues de maturité et mal socialisées, auxquelles les médias accordent une publicité assez déplacée en les présentant comme de brillants informaticiens dont les grandes entreprises en mal de sécurité se disputeraient les services à coup de super-salaires et de stock-options. Le

[5] Cette locution proverbiale m'était venue sous une forme légèrement différente, mais Christian Queinnec m'a permis de la rectifier.

fait que les pirates soient de plus en plus souvent des criminels organisés qui détournent des sommes importantes finira par avoir raison de cette idée idiote. La majorité des autres pirates sont des adolescents attardés et incompétents, qui se contentent de propager des logiciels malfaisants tout faits qu'ils n'ont eu que la peine de télécharger sur le Net.

Corollaire tout aussi idiot de cette idiotie n° 4, l'idée que les responsables de sécurité du SI devraient s'initier aux techniques de piratage : outre qu'un tel apprentissage serait pratiquement à recommencer chaque semaine, il absorberait en pure perte une énergie qui, pendant ce temps, ne serait pas consacrée à l'édification de systèmes et de réseaux sûrs *par construction*.

Idée stupide n° 5 : compter sur l'éducation des utilisateurs

Ceci semble un paradoxe : on ne reçoit jamais trop d'éducation ! Il s'agit ici de l'application de l'idée stupide n° 3 aux êtres humains : attendre que les utilisateurs aient été victimes d'un incident de sécurité et d'attaques réussies, et ensuite seulement les corriger (éduquer). En fait tout semble indiquer qu'une proportion importante d'utilisateurs seront toujours prêts, quoi qu'il advienne, à cliquer sur un lien qui promet une image pornographique ou de l'argent facile ; la nature humaine est ainsi faite, on ne la corrigera pas. Il faut donc arriver à la conclusion suivante :

Si votre politique de sécurité repose sur l'éducation des utilisateurs, alors elle est vouée à l'échec.

D'ailleurs, l'idée que l'on puisse corriger chez l'homme la propension à commettre les actes évoqués ici est une idée encore plus détestable que l'insécurité des systèmes d'information. Mieux vaut donc configurer le système de sorte que :

1. les choses dangereuses ne parviennent pas aux utilisateurs ;
2. lorsque certaines choses dangereuses passent à travers les mailles du filet (il y en aura), les conséquences en seront limitées, détectées puis contrôlées.

Cela étant dit, il faut, bien sûr et dans la mesure du possible, faire l'éducation des utilisateurs.

Idée stupide nº 6 : l'action vaut mieux que l'inaction

M. Ranum vise ici, plutôt que l'action, l'activisme. Il est clair que le responsable de site qui veut toujours adopter avant tout le monde les plus récentes technologies s'expose davantage à des incidents de sécurité que l'administrateur prudent qui attend deux ans la stabilisation du système et les retours d'expérience avant de l'implanter. En outre, pendant ce délai le coût induit par le déploiement aura probablement diminué.

On peut aussi citer l'aphorisme suivant : « Il est souvent plus facile de ne pas faire quelque chose d'idiot que de faire quelque chose d'intelligent » (attribué abusivement à l'*Art de la guerre* de Sun Tzu).

Quelques idioties de seconde classe

M. Ranum énumère pour finir quelques assertions et pratiques stupides de moindre ampleur :

- « Nous ne sommes pas une cible intéressante » : or, tout le monde est visé, les vers et les virus ne sont pas capables d'identifier les cibles qui en valent la peine ;
- « en déployant < *mettre ici le nom de votre système ou pare-feu préféré* > nous serons protégés » : non, le système ou le pare-feu qui protège, c'est celui pour lequel il y a sur le site un ingénieur (oui, un ingénieur, les gens qui savent faire ça sont des ingénieurs) compétent, qui le connaît bien et qui consacre beaucoup de son temps à s'en occuper ;
- « pas besoin de pare-feu, notre système est sûr » : non, même avec un système sûr, sans pare-feu toute application réseau est une cible facile ;
- « pas besoin de sécuriser le système, nous avons un bon pare-feu » : non, le trafic légitime qui franchit le pare-feu comporte des risques ;
- « démarrons la production tout de suite, nous sécuriserons plus tard » : non, ce ne sera jamais fait, et si cela doit l'être, cela prendra beaucoup plus de temps et de travail que de l'avoir fait au départ ;
- « nous ne pouvons pas prévoir les problèmes occasionnels » : si, vous pouvez ; prendriez-vous l'avion si les compagnies aériennes raisonnaient ainsi ?

Les cinquante prochaines années, selon Alan Cox

Alan Cox est un des principaux développeurs du noyau Linux, qu'il a notamment contribué à doter de la capacité de préemption. Il est intéressant de relever ce qu'il considère comme des facteurs de progrès de la sécurité des systèmes informatiques, en partant de son jugement sur la situation actuelle d'insécurité, qu'il estime insoutenable :

1. l'essor des systèmes de vérification de code (cf. page 98), et surtout de leur utilisation ;
2. l'amélioration des méthodes de développement, avec des langages comme Java qui règlent la majeure partie des problèmes d'allocation mémoire, principale source de failles comme l'on sait (voir page 92) ;
3. une gestion plus fine et plus restrictive de l'attribution des privilèges aux utilisateurs ;
4. les techniques de *défense en profondeur* (cf. page 13) se répandent : ainsi, le choix d'adresses aléatoires (ou plutôt imprévisibles) pour l'implantation des objets en mémoire, le verrouillage par le matériel ou par le logiciel de certaines régions de mémoire rendues non exécutables, l'usage de systèmes sécurisés comme *SELinux* (une version blindée de Linux), etc.

Détection d'intrusion, inspection en profondeur

C'est ici encore à Marcus J. Ranum[6] que nous ferons appel pour discuter la question de l'*inspection en profondeur* ; dans l'article que nous évoquons ici et dont nous retraçons les grandes lignes, il entreprend de démontrer la supériorité du mandataire applicatif sur les différents systèmes de détection et de prévention des attaques. Cet article se situe dans la droite ligne de celui que nous avons présenté au début de ce chapitre[7], en cela il défend les principes des *méthodes de sécurité a priori*, ou par construction, par opposition aux méthodes de sécurité *a posteriori*, ou curatives.

[6] http://www.ranum.com/security/computer_security/editorials/deepinspect/
[7] http://www.ranum.com/security/computer_security/editorials/dumb/

Pare-feu à états

Nous avons vu, à la section consacrée aux pare-feu (page 125), que les techniques traditionnelles de filtrage n'étaient plus suffisamment efficaces pour bloquer les attaques modernes perfectionnées, et que les pare-feu modernes utilisaient de plus en plus les techniques de suivi de connexion, qui consistent à garder en mémoire une séquence de paquets de façon à en faire l'analyse longitudinale, ce qui permet de détecter certaines malfaisances subtiles, notamment par la défragmentation de datagrammes IP et le ré-assemblage de segments TCP. Les pare-feu qui utilisent cette méthode, inaugurée en 1993 par la firme *Checkpoint*, sont appelés *stateful firewalls*, ou pare-feu à états.

Détection et prévention d'intrusion

La vague suivante de produits de sécurité fut celle des systèmes de détection et de prévention d'intrusion, dont le modèle libre est le logiciel *Snort*. Ces logiciels utilisent une base de données de signatures de vers et d'autres logiciels malfaisants, un peu à la manière d'un antivirus, et se sont révélés relativement efficaces contre la grande épidémie de vers des années 2001 à 2004, mais leur vogue décline au fur et à mesure que leur efficacité diminue. La base de signatures de *Snort* contient les descriptions de plus de 3 000 attaques.

Inspection en profondeur

Une autre voie, illustrée par certains fournisseurs (Checkpoint, Netscreen), est le pare-feu à inspection en profondeur de paquets. Il s'agit en fait d'un pare-feu à états auquel on aurait greffé la base de signatures d'un système de prévention d'intrusions, et en outre quelques procédures de détection d'anomalies protocolaires.

Critique des méthodes de détection

Dans son article cité en référence, Marcus J. Ranum cite en exemple de procédure d'inspection en profondeur l'analyse du protocole SMTP par le logiciel NFR : ce logiciel examine la séquence complète de commandes SMTP du début à la fin de l'envoi de message, et émet une alerte en cas d'occurrence d'une commande `Mail From :` émise par le logiciel client avant la commande `RCPT To :` correspondante ; une telle analyse est très efficace, parce qu'un logiciel de messagerie de

bonne foi n'utilisera *jamais* une telle séquence, et qu'il ne peut s'agir que d'une anomalie, d'une tentative de piraterie. Mais, dans cet exercice, un pare-feu à base de mandataire applicatif sera supérieur à un système de détection d'anomalies protocolaires, parce que par définition le mandataire *exécute* le protocole, et que de ce fait aucune anomalie ne peut lui échapper. Alors que le système de détection en est réduit à *supputer* ce que le protocole exécute, le mandataire *est* l'implantation du protocole.

Comme un mandataire applicatif exécute les séquences protocolaires pour lesquelles il a été programmé dès sa conception, il n'accomplit, par construction, que des actions autorisées, il réalise le principe « par défaut, tout est interdit ».

Le logiciel de détection d'attaques examine sa base de données de signatures d'attaques, et s'il ne trouve aucune signature qui corresponde à la séquence examinée, il considère qu'elle est légitime, ce qui réalise le principe « par défaut, tout est permis ».

Face à des profils d'attaques de plus en plus nombreux, de plus en plus divers et de plus en plus complexes, nous pensons que dans la course aux armements entre attaquants et systèmes de détection, les attaquants submergeront tôt ou tard les défenseurs, et nous nous rangerons à l'avis de Marcus J. Ranum : l'avenir est au mandataire applicatif.

À qui obéit votre ordinateur ?

En ce début de siècle obsédé par des menaces contre la sécurité et l'ordre public se manifestent des tendances au renforcement du contrôle social, qui, dans le domaine qui nous intéresse ici, se traduisent par de vastes projets de surveillance des usages des ordinateurs et des réseaux, et d'interdiction de ceux de ces usages qui ne reçoivent pas l'assentiment des puissances à l'œuvre dans l'industrie des médias, par exemple pour ce qui touche à la diffusion et à l'échange de musique et de films par l'Internet. Ces tendances répressives constituent un danger parce que, comme toutes les mesures excessives et abusives, elles se retournent contre leur objectif initial : elles se révèlent nuisibles à la disponibilité et à la liberté d'usage légitime des systèmes d'information, tout en concentrant un pouvoir excessif dans les mains d'un petit nombre d'entreprises privées.

Conflit de civilisation pour les échanges de données numériques

L'ubiquité de l'informatique et de l'Internet jusque dans les habitudes domestiques et culturelles a engendré de nouveaux comportements dans la vie privée des citoyens, au nombre desquels la publication de sites Web privés tels que les blogs, l'échange de conversations et de documents de toute sorte par le réseau, qu'il s'agisse de textes, d'images ou de sons, ainsi que de nouvelles formes de créativité, puisque tel qui était mauvais dessinateur au fusain et au canson peut se révéler brillant graphiste électronique, et tel autre qui souffrait du symptôme de la page blanche avec un stylo frise la graphomanie avec un clavier et un écran. Ces nouvelles pratiques culturelles sont souvent associées à l'usage de logiciels libres, ou fécondées par eux. Elles ont considérablement élargi le champ de la liberté d'expression, et apparaissent comme une évolution majeure de la civilisation et de la culture.

Cette véritable révolution culturelle rencontre l'hostilité des industriels de la culture ; rappelons ici quelles sont les grandes puissances de cette industrie : le marché mondial de l'édition numérique (CD et DVD) est contrôlé par quatre géants, les « majors », EMI, Sony, TimeWarner et Universal. Ces industriels de la culture, au lieu de s'adapter à ces évolutions en imaginant de nouvelles formes de commerce, comme Amazon a bien su le faire, ont préféré s'engager dans un combat conservateur (perdu d'avance) pour préserver leurs rentes, assises sur des technologies vieillissantes vendues à des tarifs surévalués, et faire interdire les nouvelles pratiques culturelles évoquées ci-dessus. À cette fin ils se sont engagés dans un combat juridico-technique planétaire pour faire adopter par les États des législations prohibitionnistes à l'encontre des nouvelles pratiques de création et d'échange, qui reposent sur les ordinateurs et le réseau.

Le combat juridique se double d'un combat technique. En fait, l'offensive des majors avance sur deux fronts :

- créer des dispositifs techniques destinés à empêcher ou à surveiller les pratiques jugées indésirables par les majors ; nous avons eu l'occasion de décrire un de ces procédés à la page 58 ; dans cette entreprise ils reçoivent le soutien de certains industriels de l'informatique, notamment Intel et Microsoft ;
- faire adopter par les États des législations qui interdisent le contournement de ces dispositifs techniques, et qui permettraient de punir les pratiques désapprouvées par les majors.

Cette combinaison de dispositions techniques et légales devrait être verrouillée, si les rêves des majors se réalisent, par l'adoption en Europe d'une législation sur la brevetabilité du logiciel, inspirée de celle qui a cours aux États-Unis, et qui pourrait empêcher la création de logiciels libres, notamment dans ce domaine de la création et de la diffusion d'œuvres de l'esprit. Dans ce combat des brevets logiciels, les majors ont reçu le renfort de Siemens, Nokia, Philips et Alcatel. Autant dire que les forces hostiles aux nouvelles pratiques culturelles disposent de moyens économiques et de pouvoirs d'influence considérables.

Dispositifs techniques de prohibition des échanges

Gestion des droits numériques (DRM)

Nous avons déjà évoqué à la page 58 le protocole de gestion des droits numériques DRM, en l'occurrence pour en signaler une réalisation fautive et frauduleuse. DRM vise à protéger des données numériques enregistrées sur CD ou DVD, ou diffusées par le réseau, au moyen d'un système de chiffrement et de signature. Le fichier numérique qui contient, par exemple, le film ou la musique est chiffré et compressé. Il ne pourra être lu qu'au moyen d'un logiciel spécial, qui sera éventuellement fourni avec le fichier et installé sur le même support. Pour lire le fichier, c'est-à-dire voir le film ou écouter la musique, l'acheteur devra fournir une clé secrète qui lui aura été remise au moment du paiement. Le logiciel DRM pourra également, au gré du vendeur, limiter le nombre de copies possibles du fichier, ou le nombre de lectures, ou la date limite de lecture.

Un des multiples inconvénients du protocole DRM, c'est qu'il limite l'usage légitime des données qu'il protège : si le logiciel de lecture ne fonctionne que sur tel ou tel modèle de lecteur de DVD ou avec tel ou tel système d'exploitation, les propriétaires de systèmes différents ne pourront pas utiliser le DVD en question, quand bien même ils l'auront payé, et la loi sur les brevets logiciel leur interdira de créer un logiciel libre destiné à résoudre ce problème. La situation décrite ici n'est pas du tout un cas d'école, elle s'est effectivement produite ; ainsi le Norvégien Jon Johansen a été poursuivi en 2000 par les tribunaux de son pays, à la demande de l'Association américaine pour le contrôle de la copie de DVD (DVD-CAA), pour le simple fait d'avoir tenté de lire ses propres DVD, et d'avoir écrit pour ce faire le logiciel DeCSS pour le décodage des DVD sous Linux ; il a finalement été

acquitté en 2003. Et on ne compte plus les acheteurs dépités de ne pas pouvoir lire leur DVD tout neuf sur leur lecteur tout neuf, grâce à DRM.

Trusted Computing Platform Alliance (TCPA)

Trusted Computing Platform Alliance est une association d'entreprises d'informatique (HP, IBM, Intel, Microsoft...) qui se donne pour objectif la sécurité des équipements et des réseaux informatiques, et qui développe pour cela des dispositifs matériels et logiciels qu'elle souhaite incorporer au cœur des ordinateurs et des systèmes d'exploitation de demain. Ce qui est à noter, c'est que les dispositifs envisagés par TCPA sont destinés à être implantés dans des couches basses du matériel et du logiciel, de telle sorte que l'utilisateur ne pourra pas intervenir pour modifier leur comportement.

Le principe des dispositifs TCPA consiste à attribuer une signature à chaque élément de système informatique (logiciel, document), et à déléguer à un tiers de confiance la possibilité de vérifier si l'objet considéré peut être légitimement utilisé sur le système informatique local.

Tout élément non signé ou dont la signature n'est pas agréée par le tiers de confiance sera rejeté. On imagine les applications d'un tel dispositif à la lutte contre les virus. Mais aussi, si par exemple le « tiers de confiance » est le fournisseur du système (et qui pourra l'empêcher de s'arroger cette prérogative ?), il lui sera possible de vérifier que les applications utilisées sont bien conformes au contrat de licence concédé à l'utilisateur. Un des problèmes soulevés par cette technique est que l'utilisateur final perd ainsi toute maîtrise de ce qui peut ou ne peut pas être fait avec son propre ordinateur. C'est par ce procédé, notamment, qu'Apple s'assure que son système d'exploitation Mac OS X ne peut être exécuté que sur les ordinateurs à processeur Intel de sa fabrication. Mais on pourrait imaginer que cette méthode soit utilisée pour empêcher l'usage de certains logiciels libres.

Les spécifications émises par TCPA formulent la définition du *Trusted Platform Module* (TPM), destiné à procurer des *primitives de sécurité* dans un environnement sûr. Par « primitives » on entend : signature électronique, génération de nombres pseudo-aléatoires, protection de la mémoire, accès à un état garanti de l'information contenue par le TPM. L'intégrité et l'authenticité de ces primitives et de leur exécution sont assurées par des dispositifs matériels. Le TPM doit être

un composant discret, identifiable de façon distincte sur la carte-mère de l'ordinateur, mis en œuvre au moyen d'un pilote activé par le BIOS. Ces dispositions assurent l'indépendance du fonctionnement du TPM à l'égard de ce qui se passe dans le système accessible à l'utilisateur. Par exemple, l'utilisation du TPM peut garantir qu'un dispositif de DRM n'aura pas été modifié ou contourné par un utilisateur, opération triviale avec les dispositifs de DRM actuels, implantés purement en logiciel.

Next-generation secure computing base (NGSCB)

Next-generation secure computing base est le nom d'un projet Microsoft antérieurement baptisé Palladium. NGSCB devait être utilisé par Microsoft pour implanter une architecture de confiance dans son système le plus récent, *Vista*, mais cette installation est différée *sine die*, sans doute à cause des réticences suscitées par les aspects *Big Brother* prêtés au système.

Avec NGSCB, qui fonctionne à l'aide d'un processeur cryptographique, le système d'exploitation *Vista* travaillera dans un environnement de sécurité. Les principes en sont d'incorporer la cryptographie au système d'exploitation pour garantir l'intégrité des échanges entre processus, entre les processus et la mémoire, entre les processus et les disques, et entre les processus et les dispositifs d'entrée-sortie (clavier, souris, écran...).

Ce mode de fonctionnement permettrait de vérifier que des fichiers créés par une application ne peuvent être lus ou modifiés que par cette même application ou par une autre application autorisée, de protéger le système contre l'exécution de codes non autorisés tels que les virus et tout programme non autorisé par l'utilisateur ou l'administrateur, et de mener à bien l'édification de systèmes informatiques vraiment distribués dont chaque composant puisse faire confiance aux autres parties du système (logicielles ou matérielles) même si celles-ci font partie d'un système distant.

Les détracteurs du projet ne manquent pas d'observer qu'avec NGSCB Microsoft aura les moyens d'exercer un contrôle total sur les ordinateurs de ses clients, et notamment d'y persécuter les logiciels libres qu'il estimerait contraires à ses intérêts. Un tel dispositif sera aussi de nature à accroître l'efficacité des systèmes de DRM... et à aggraver les abus qui en découlent, signalés ci-dessus.

Les développements récents de cette politique de contrôle des usages sont évoqués par la revue *Microprocessor Report* [70] : les industriels prévoient de lancer une offre de diffusion vidéo haute définition à la demande par l'Internet, qui sera encadrée par des mesures techniques de protection drastique, en l'occurrence les plates-formes matérielles *Viiv* d'Intel ou *Live !* d'AMD, le procédé de chiffrement HDCP *(High Bandwidth Digital Content Protection)* et le dispositif de connexion HDMI *(High Definition Multimedia Interface)*. Tout cela signifie qu'il faudra, pour accéder à cette offre, faire l'emplette d'un nouvel ordinateur et d'un nouveau système d'exploitation, et que les systèmes libres tels que Linux ou OpenBSD en seront probablement exclus.

Informatique de confiance, ou informatique déloyale ?

Richard M. Stallman a écrit un article[8] de critique de ces projets qui prétendent nous mener vers une informatique « de confiance », où il la qualifie, au contraire, d'*informatique déloyale*. La déloyauté réside dans les possibilités que NGSCB offre au « tiers de confiance » pour agir sur les données stockées par l'ordinateur, à l'insu de l'utilisateur légitime et sans que celui-ci puisse rien faire pour l'empêcher. R.M. Stallman donne des exemples d'actions déloyales rendues possibles par de telles techniques :

« Rendre impossible le partage des fichiers vidéos et musicaux est une mauvaise chose, mais cela pourrait être pire. Il existe des projets pour généraliser ce dispositif aux messages électroniques et aux documents – ayant pour résultat un e-mail qui disparaîtrait au bout de deux semaines, ou des documents qui pourront seulement être lus sur les ordinateurs d'une société mais pas sur ceux d'une autre. (...)

Les logiciels de traitement de texte tels que Word de Microsoft pourraient employer « l'informatique déloyale » quand ils enregistrent vos documents, pour s'assurer qu'aucun autre traitement de texte concurrent ne puisse les lire. (...)

Les programmes qui utilisent « l'informatique déloyale » téléchargeront régulièrement de nouvelles règles par Internet, et imposeront ces règles automatiquement à votre travail. »

[8] `http://www.gnu.org/philosophy/can-you-trust.fr.html`

Nous trouvons sur le site de l'Adullact[9] une analyse comparative des licences logicielles, qui corrobore les craintes que l'on peut avoir à l'égard des mesures techniques de protection associées à la gestion des droits numériques :

« L'intégration de la gestion des droits numériques (DRM) dans *Windows* implique que la société *Microsoft* peut à tout moment révoquer votre droit d'accès aux contenus sécurisés si elle considère votre logiciel compatible-DRM compromis. Une liste de logiciels révoqués est automatiquement installée sur votre ordinateur à chaque téléchargement de contenus sécurisés. Une mise à jour de votre logiciel compatible-DRM est alors nécessaire pour continuer à accéder à vos fichiers sécurisés. Cette révocation n'empêche cependant pas l'accès à des contenus non protégés par les DRM. »

De tels projets constituent effectivement une menace contre la liberté d'expression, et contre les libertés publiques en général. Les entreprises qui les fomentent exploitent abusivement pour leur promotion la psychose de sécurité consécutive au 11 septembre 2001. La puissance de ces entreprises semble considérable, mais nous pensons qu'elles seront néanmoins impuissantes à endiguer les nouvelles pratiques culturelles, parce que celles-ci sont déjà le fait de plusieurs dizaines de millions d'internautes de par le monde, qu'il s'agisse de la publication et de l'échange sur Internet ou du recours aux logiciels libres.

Mesures de rétorsion contre les échanges de données

Le gouvernement français a demandé à Antoine Brugidou, d'*Accenture*, et à Gilles Kahn, alors président de l'INRIA, un rapport (disponible en ligne) sur les échanges de fichiers musicaux par Internet et sur les moyens éventuels de les contrôler ou de les bloquer par des dispositifs techniques[10]. Ce rapport est destiné notamment à répondre aux préoccupations des syndicats français des entreprises de l'édition phonographique et cinématographique, qui ne pensent qu'à interdire, détecter, bloquer et punir les échanges en question.

La réponse donnée par le rapport est que les mesures techniques de contrôle et d'interdiction dont rêvent les éditeurs seront difficiles à mettre en œuvre, très coûteuses, et d'une efficacité limitée dans le temps. En effet les systèmes de filtrage de flux sur l'Internet, pour être réellement efficaces, devraient être installés au cœur

[9] `http://www.adullact.org/documents/comparatif_licences.html`
[10] `http://www.recherche.gouv.fr/discours/2005/musiqueinternet.htm`

des réseaux des fournisseurs d'accès à l'Internet (FAI) ; or les FAI ne manifestent aucun enthousiasme à l'idée d'encombrer leurs infrastructures avec ces matériels onéreux, qui vont ralentir le débit de leurs réseaux, et dont l'objectif est d'empêcher leurs clients de se livrer aux activités pour lesquelles justement ils ont souscrit un abonnement à haut débit. Si le ministère de la Culture soutient les syndicats d'éditeurs, le ministère de l'Industrie soutient les FAI.

Les principes de fonctionnement des dispositifs de filtrage ne sont pas déterministes, mais heuristiques : en effet rien ne permet de distinguer de façon sûre un échange poste à poste « suspect » d'un autre type de trafic, comme nous l'avons vu à la page 210. Le filtrage des adresses IP, des numéros de ports ou d'autres données de protocole sont totalement inefficaces contre ce type de trafic. Les systèmes de détection doivent donc reconnaître la « signature » d'un échange, puis ouvrir les paquets pour en investiguer le contenu et mettre en évidence le « délit ». Chaque fois que le protocole (non public) du système poste à poste sera modifié, les systèmes de détection seront mis en échec.

Les sociétés *Allot* et *Cisco* (gamme *P_Cube*) proposent des solutions de filtrage de protocole basés sur la reconnaissance de signature. Les sociétés *Audible Magic* (boîtier *CopySense*) et *Advestigo* proposent du filtrage de contenus. Il y a aussi des systèmes de filtrage sur le poste client, qui supposent la collaboration de l'utilisateur, par exemple dans le cas de parents qui souhaitent empêcher leurs enfants de s'adonner au téléchargement ; cet état de fait pourrait changer avec des dispositifs tels que TCPA (cf. page 235) et NGSCB (cf. page 236), susceptibles d'être utilisés pour surveiller un ordinateur sans le consentement de son propriétaire légitime, mais nous voulons croire que des situations aussi iniques ne pourront pas voir le jour.

MM. Brugidou et Kahn envisagent dans leur rapport plusieurs scénarios de déploiement d'outils de filtrage sur les infrastructures des FAI. Ces équipements, pour jouer leur rôle, devront être installés en coupure, ce qui signifie qu'ils devront être adaptés au débit des infrastructures, soit aujourd'hui généralement 1 gigabit/s, mais bientôt 10 Gb/s, c'est-à-dire qu'ils seront coûteux. Ainsi, le rapport envisage une solution suggérée par un syndicat professionnel et adaptée au réseau de France Télécom : elle consisterait à implanter un boîtier *Allot* à 1 Gb/s en coupure derrière chaque BAS *(Broadband Access Server)* du réseau ADSL de l'opérateur, soit à l'époque de l'étude 143 boîtiers. Le prix de chaque boîtier est de plusieurs dizaines de milliers d'euros. Les fournisseurs d'accès à l'Internet n'ont

guère d'attrait pour ce type d'investissement, qui pénaliserait surtout leurs clients en termes de performances du réseau et de liberté d'usage de leurs ordinateurs, et ce pour une efficacité très discutable.

> **VOCABULAIRE BAS et DSLAM**
>
> Les accès ADSL *(Asymmetric Digital Subscriber Line)* d'un opérateur sont raccordés à un DSLAM *(DSL Access Multiplexer)*. Un DSLAM sera en général installé dans un central (nommé désormais *Nœud de Raccordement d'Abonné*, ou NRA) et desservira une zone de 4 ou 5 km de rayon. Un BAS concentrera le trafic d'une dizaine de DSLAM.

La voie du blocage des échanges de fichiers sur le réseau semble donc peu prometteuse pour les industriels de la culture : on comprend qu'ils soient tentés de se rabattre sur l'implantation de la gestion numérique des droits au cœur de l'ordinateur, avec des technologies telles que les TPM (cf. page 235) et NGSCB (cf. page 236). Ces dernières solutions pourraient être techniquement efficaces, mais elles seraient inacceptables pour les utilisateurs, qui y verraient un empiètement intolérable sur leur liberté d'utiliser comme bon leur semble les objets et les supports numériques qu'ils ont achetés.

Gestion des droits numériques (DRM) et politique publique

Dans un article des *Communications of the ACM (CACM)* de juillet 2005[11], Edward W. Felten, professeur d'informatique et de politique publique à l'université de Princeton, où il dirige en outre le *Center for Information Technology Policy*, a proposé aux instigateurs et aux auteurs de politiques publiques pour la gestion des droits numériques six principes qui lui semblent s'imposer :

- **Pluralité et concurrence :** une politique publique des droits numériques devrait permettre la pluralité des systèmes de gestion de droits, et promouvoir l'interopérabilité entre ces systèmes.
- **Équilibre du droit d'auteur :** les législations relatives au droit d'auteur ont, traditionnellement, cherché un équilibre entre la rémunération de l'auteur et le droit d'accès du public ; la gestion des droits numériques et les législations qui s'y appliquent devraient respecter cet équilibre, non le remettre en cause.

[11] Cf. `http://www.csl.sri.com/users/neumann/insiderisks05.html#181`

- **Protection du consommateur :** les systèmes de gestion des droits numériques ne devraient pas restreindre les droits des consommateurs, et les politiques publiques qui s'y appliquent devraient les protéger.
- **Protection de la vie privée :** les politiques publiques relatives à la gestion des droits numériques devront veiller à la protection de la vie privée, en empêchant que les systèmes de gestion de ces droits ne deviennent des moyens d'espionner les utilisateurs en recueillant des données sur leurs comportements et leurs pratiques culturelles ou autres.
- **Recherche et débat public :** la politique publique devra favoriser la recherche et le débat public sur les questions relatives aux droits numériques, et faire obstacle aux dérives récentes, qui ont vu certaines entreprises tenter d'utiliser les législations sur la propriété intellectuelle pour assigner en justice des auteurs d'articles scientifiques ou de logiciels de recherche.
- **Délimitation du champ d'application :** les politiques publiques devront voir leur champ d'application délimité précisément au domaine où elles seront utiles, et éviter les formulations susceptibles d'être détournées par des avocats trop habiles à l'encontre d'usages légitimes des droits numériques.

Ces principes équilibrés devraient pouvoir recueillir l'assentiment de tous les interlocuteurs de bonne foi dans le débat. Si la gestion des droits numériques devait devenir soit, pour l'industrie culturelle, un moyen de tondre plus efficacement un consommateur sans défense, soit, pour une mouvance libertaire extrémiste, un moyen de profiter des œuvres d'art sans rémunérer les artistes, elle serait de toutes les façons condamnée à l'échec ; nous croyons que ces deux voies extrêmes n'ont aucun avenir.

Conclusion

Le proverbe dit « mieux vaut prévenir que guérir » : au terme du parcours des divers aspects de la sécurité des systèmes d'information, nous pourrions presque dire qu'en ce domaine prévenir est impératif, parce que guérir est impossible et de toute façon ne sert à rien. Lorsqu'un accident ou un pirate a détruit les données de l'entreprise et que celle-ci n'a ni sauvegarde ni site de secours, elle est condamnée, tout simplement : les personnels de ses usines ne savent plus quoi produire ni à quels clients livrer quoi, ses comptables ne peuvent plus encaisser les factures ni payer personnels et débiteurs, ses commerciaux n'ont plus de fichier de prospects.

Un accident moins grave aura sans doute des conséquences moins radicales, mais en règle générale les conséquences d'un incident de sécurité sont irréversibles si aucune prévention n'avait été organisée avant qu'il n'advienne.

Il est impossible de connaître à l'avance tous les types de menaces et de détecter toutes les vulnérabilités, puisque, ainsi que nous l'avons signalé et répété, il en apparaît de nouvelles chaque semaine, par dizaines. Par conséquent, l'analyse de risques se révèle vite une aporie si on lui accorde trop de confiance, si on la croit déterministe ; il convient de s'y adonner, mais avec scepticisme.

L'analyste de risques sceptique sera un responsable de sécurité agnostique et pessimiste : il *sait* que son pare-feu sera franchi, que son antivirus ne sera pas à jour, que son système de détection d'intrusion ne le préviendra pas de l'attaque, que ses copies de sauvegarde seront corrompues, que son site de secours sera inondé ou détruit par un incendie, que son système redondant ne se déclenchera pas ;

mais, éduqué dans la religion probabiliste, il *sait* que toutes ces catastrophes ne surviendront pas simultanément.

L'idée de *défense en profondeur* n'est pas sans parenté avec la démarche agnostique probabiliste, mais elle s'en distingue : si la garnison de mon pare-feu est finalement submergée par l'assaillant, elle en aura néanmoins réduit les effectifs avant de succomber, ce qui facilitera la mission des escadrons d'antivirus, et ainsi mon système redondant risquera moins d'être saboté par un ver qui pourrait l'empêcher de se déclencher. Si au contraire je mise tout sur mon pare-feu ou sur mon réseau privé virtuel et que derrière cette protection je commets des imprudences, je succombe au syndrome de la *ligne Maginot*, le jour où la défense est enfoncée tout est perdu. Or, si une chose est sûre, c'est que la défense sera enfoncée. Un jour.

Une autre certitude, c'est que le risque ne vient pas seulement de l'extérieur, les sources de danger prolifèrent aussi à l'intérieur du réseau, et d'ailleurs la frontière entre l'intérieur et l'extérieur tend non pas à disparaître, mais à devenir poreuse et floue, avec les systèmes mobiles en tout genre qui entrent et qui sortent, les tunnels vers d'autres réseaux, les nouveaux protocoles infiltrables et furtifs. Les protocoles de téléphonie par Internet, de visioconférence et autres systèmes multimédia sont *tous* des failles béantes de sécurité, et la situation sur ce front ne s'améliorera pas avant des années.

Nous voyons que les menaces sont protéiformes, les vulnérabilités foisonnantes et le tout en transformation constante : c'est dire que le responsable de sécurité ne choisit pas le terrain sur lequel il va devoir manœuvrer, il va lui falloir faire preuve d'adaptabilité et de pragmatisme. S'il ne veut pas se trouver condamné à réagir frénétiquement mais trop tard à des avalanches d'incidents mystérieux, il devra néanmoins établir un socle stable pour son activité, dont nous avons établi en principe qu'elle sera essentiellement préventive. Pour cela il lui faudra principalement deux choses : une vraie compétence technique dans son domaine, suffisamment large et profonde pour embrasser réseaux et systèmes, et, au sein de son entreprise, le pouvoir d'édicter les règles dans son domaine, et de les faire respecter : interdire les protocoles dangereux, imposer la mise à jour automatique des antivirus, mettre son veto à tel ou tel passe-droit dans le pare-feu. Cela s'appelle une politique de sécurité.

Cette compétence technique et son instanciation dans une politique de sécurité, il serait vain d'espérer en faire l'économie en lui substituant des procédures. Il

ne manque pas de méthodes qui laissent croire que que la sécurité des systèmes d'information pourrait être assurée par des routines administratives : nous avons signalé et expliqué leur vanité à la fin du premier chapitre de ce livre. Nous dirons que ces méthodes de sécurité sont procédurales, ou, plus crûment, qu'elles sont bureaucratiques.

Nous avons donc le choix entre ces méthodes bureaucratiques et celles que nous appellerons méthodes de sécurité négative, parce qu'elles proposent de colmater les failles dès que celles-ci sont découvertes et d'interdire les malfaisances après qu'elles se sont manifestées : ni celles-là ni celles-ci ne sont satisfaisantes, nous l'avons vu. Nous préconiserons plutôt celles qui visent ce que nous appellerons la *sécurité positive*, parce qu'elles posent *a priori* ce qui est sûr, et qu'elles établissent la sécurité à la conception des systèmes, par la définition de ce qu'ils doivent faire et l'interdiction du reste selon une règle que nous énoncerons ainsi : « n'est permis que ce qui est explicitement permis, tout le reste est interdit ».

Par exempe, à l'heure où pratiquement toutes les applications informatiques sont fondées sur les techniques du Web, nous pensons, en suivant Marcus J. Ranum, qu'un outil de choix pour la sécurité positive est le *mandataire applicatif (reverse proxy)* : il s'agit d'un serveur Web spécialisé, qui reçoit les messages du protocole HTTP, les filtre, rejette ce qui n'est pas autorisé et *réécrit* les requêtes avant de les transmettre au « vrai » serveur, ce qui élimine tout imprévu, et notamment toute une famille d'attaques par injection de code. Cette méthode revient à écrire sa propre version du protocole, adaptée exactement à ce que l'on veut faire.

De façon générale, l'évolution de l'informatique, de ses usages, et par conséquent des systèmes d'information, est déterminée par l'offre de technologie plus que par les demandes des utilisateurs, parce que celle-là évolue plus vite que celles-ci. Pour des raisons évidentes, c'est encore plus vrai pour les questions de sécurité, parce que les utilisateurs ne « demandent » rien, et que l'« offre » est par définition destinée à surprendre ses « clients » par des attaques auxquelles ils ne s'attendent pas. La lutte contre cette « offre » un peu spéciale ne peut donc reposer sur les attentes du client, et la veille technologique « tous azimuts », si elle est nécessaire, ne saurait prétendre à l'efficacité totale. Ce qui renforce l'argument pour la sécurité positive.

Pour toutes les raisons qui viennent d'être énoncées, nous pouvons conclure en disant avec Bruce Schneier [100] que la sécurité du système d'information n'est pas

et ne peut pas être contenue dans un dispositif ni dans un ensemble de dispositifs, qu'elle ne peut pas non plus être contenue dans les limites temporelles d'un *projet*, mais qu'elle est un *processus* ou, si l'on veut, une *activité*. Nous entendons par là que les ingénieurs de sécurité du SI doivent se consacrer à cette activité, pas forcément à plein temps, mais en permanence, sur plusieurs fronts : veille scientifique et technologique, surveillance des journaux d'événements, audit des infrastructures et des applications, sensibilisation et formation des utilisateurs, expérimentation de nouveaux outils et de nouveaux usages. La démarche de sécurité doit être active : la détection des failles et des attaques, et les réponses qui leur sont données, ne sont pas suffisantes, mais elles sont nécessaires, parce qu'avec l'ubiquité de l'Internet nous sommes entrés dans une ère où le régime de menaces est de basse intensité, mais les menaces sont permanentes. Il faut savoir que parmi ces menaces certaines se réaliseront, qu'il faut s'y préparer et apprendre à leur survivre, ce qui suppose que l'on y ait pensé *avant*.

Bibliographie

[1] « MIT researchers uncover mountains of private data on discarded computers ». *Massachusetts Institute of Technology, News Office*, 15 janvier 2003. `http://web.mit.edu/newsoffice/2003/diskdrives.html`.

[2] « Site de l'Adullact ». *Association des développeurs et des utilisateurs de logiciels libres pour l'administration et les collectivités territoriales*, 2005. `http://www.adullact.org/documents/comparatif_licences.html`.

[3] « Site de l'OSSIR ». *Observatoire de la sécurité des systèmes d'information et des réseaux*, 2005. Cette association est aujourd'hui le meilleur cénacle francophone dans son domaine. `http://www.ossir.org`.

[4] « Site du journal MISC ». *MISC*, 2005. Revue francophone de sécurité informatique. `http://www.miscmag.com`.

[5] « Sécurité de Perl ». *Site Perl de l'ENSTIMAC*, 23 mars 2006. `http://perl.enstimac.fr/DocFr/perlsec.html`.

[6] Jean-François Abramatic. « Croissance et évolution de l'Internet ». Dans *Université de tous les savoirs – Les Technologies*, volume 7, Paris, 2002. Odile Jacob.

[7] Jean-Raymond Abrial. *The B Book - Assigning Programs to Meanings*. Cambridge University Press, Cambridge, 1996.

[8] Pascal Aubry, Julien Marchal, et Vincent Mathieu. « *Single Sign-On Open Source* avec CAS ». 2003. `http://2003.jres.org/actes/paper.139.pdf`.

[9] Autorité de régulation des communications électroniques et des postes (ARCEP). « Le cadre réglementaire des réseaux RLAN / Wi-Fi depuis le 25 juillet 2003 », 8 août 2003. `http://www.art-telecom.fr/dossiers/rlan/schema-rlan.htm`.

[10] Gildas Avoine, Pascal Junod, et Philippe Oechslin. *Sécurité informatique – Exercices corrigés*. Vuibert, Paris, 2004. Préface de Robert Longeon.

[11] Daniel Azuelos. « Architecture des réseaux sans fil ». Dans JRES, editor, *Actes du congrès JRES*, 2005. http://2005.jres.org/tutoriel/Reseaux_sans_fil.livre.pdf.

[12] Général de Brigade Bailey, MBE. « Le combat dans la profondeur 1914-1941 : la naissance d'un style de guerre moderne ». *Les cahiers du Retex*, (15), 17 mars 2005. http://www.cdef.terre.defense.gouv.fr//publications/cahiers_drex/cahier_retex/retex15.pdf.

[13] Scott Barman. *Writing Information Security Policies*. New Riders, Indianapolis, USA, 2002.

[14] Salman A. Baset et Henning Schulzrinne. « An Analysis of the Skype Peer-to-Peer Internet Telephony Protocol ». *arXiv.org*, 15 septembre 2004. http://arxiv.org/ftp/cs/papers/0412/0412017.pdf.

[15] Didier Bert, Henri Habrias, et Véronique Viguié Donzeau-Gouge (éd.). « Méthode B (numéro spécial) ». *Technique et science informatique*, 22, 1/2003.

[16] Philippe Biondi et Fabrice Desclaux. « Silver Needle in the Skype ». *BlackHat Europe*, 2-3 mars 2006. http://www.secdev.org/conf/skype_BHEU06.pdf.

[17] Laurent Bloch. *Les systèmes d'exploitation des ordinateurs – Histoire, fonctionnement, enjeux*. Vuibert, Paris, 2003. Texte intégral disponible ici : http://www.laurent-bloch.org/article.php3?id_article=13.

[18] Laurent Bloch. *Systèmes de fichiers en réseau : NFS, SANs et NAS*. 2005. Texte disponible ici : http://www.laurent-bloch.org/Livre-Systeme/livre008.html.

[19] Laurent Bloch. *Systèmes d'information, obstacles et succès – La pensée aux prises avec l'informatique*. Vuibert, Paris, 2005. Extraits et documents complémentaires disponibles ici : http://www.laurent-bloch.org/rubrique.php3?id_rubrique=5.

[20] Laurent Bloch. « Théorie et pratique de la commande publique ». 2005. http://www.laurent-bloch.org/SI-Projets-extraits/livre005.html.

[21] Frédéric Bonnaud. « Signer et chiffrer avec GnuPG ». *Lea-Linux.org*, 2005. http://lea-linux.org/cached/index/Reseau-secu-gpg-intro.html.

[22] Isabelle Boydens. *Informatique, normes et temps*. Bruylant, Bruxelles, 1999.

[23] Philippe Breton. *La tribu informatique – Enquête sur une passion moderne*. Métailié, Paris, 1991.

[24] Christophe Brocas et Jean-Michel Farin. « De la sécurité d'une architecture DNS d'entreprise ». *MISC*, (23), Janvier-février 2006.

[25] Antoine Brugidou et Gilles Kahn. « Étude des solutions de filtrage des échanges de musique sur Internet dans le domaine du *peer-to-peer* », 9 mars 2005. http://www.recherche.gouv.fr/discours/2005/musiqueinternet.htm.

[26] Franck Cappello. « P2P : Développements récents et perspectives ». Dans *6e jour-*

nées réseau JRES, 2005. En ligne ici : http://2005.jres.org/slides/152.pdf.

[27] Centre d'Expertise de Réponse et de Traitement des Attaques informatiques (Cert-RENATER). « Site du Cert-RENATER », 10 septembre 2006. http://www.renater.fr/rubrique.php3?id_rubrique=19.

[28] Centre d'Expertise Gouvernemental de Réponse et de Traitement des Attaques informatiques (CERTA). « Site du CERTA », 10 septembre 2006. http://www.certa.ssi.gouv.fr/.

[29] Centre d'Expertise Gouvernemental de Réponse et de Traitement des Attaques informatiques (CERTA). « Sécurité des réseaux sans fil (Wi-Fi) », 26 octobre 2004. http://www.certa.ssi.gouv.fr/site/CERTA-2002-REC-002/.

[30] D. Brent Chapman et Elizabeth D. Zwicky. *Firewalls – La sécurité sur l'Internet*. O'Reilly, Sebastopol, Californie (Paris pour la traduction), 1995. Traduction de Jean Zundel.

[31] Société ClearSy. « Atelier B ». juin 2004. http://www.atelierb.societe.com/.

[32] Commission Nationale Informatique et Libertés. « Norme simplifiée n° 46 », 13 janvier 2005. http://www.cnil.fr/index.php?id=1231.

[33] Computer Emergency Response Team - Coordination center. « Site du Cert-CC », 10 septembre 2006. http://www.cert.org/.

[34] Computer Emergency Response Team - Industrie, Services et Tertiaire (Cert-IST). « Site du Cert-IST », 10 septembre 2006. http://www.cert-ist.com/.

[35] Thomas Cormen, Charles Leiserson, Ronald Rivest, et Clifford Stein. *Introduction à l'algorithmique*. Dunod (pour la traduction française), Paris, 2002. Une somme d'une complétude impressionnante ; si les exposés mathématiques des algorithmes sont d'une grande clarté, le passage à la programmation (en pseudo-code) est souvent difficile.

[36] Alan Cox et Edd Dumbill. « The Next 50 Years of Computer Security : An Interview with Alan Cox ». *O'Reilly Network*, 12 septembre 2005. http://www.oreillynet.com/pub/a/network/2005/09/12/alan-cox.html.

[37] CROCUS (collectif). *Systèmes d'exploitation des ordinateurs*. Dunod, Paris, 1975. Ce manuel, quoique assez ancien, conserve un intérêt certain par sa rigueur dans l'introduction des concepts et du vocabulaire, et en a acquis un nouveau, de caractère historique, par la description de systèmes aujourd'hui disparus.

[38] Cunningham et Cunningham. « Cee Language and Buffer Overflows ». *Cunningham and Cunningham, Inc.*, 16 août 2005. http://c2.com/cgi/wiki?CeeLanguageAndBufferOverflows.

[39] Fabrice Desclaux. « Skype uncovered – Security study of Skype ». *OSSIR – Groupe sécurité Windows*, 7 novembre 2005. http://www.ossir.org/windows/

supports/2005/2005-11-07/EADS-CCR_Fabrice_Skype.pdf.

[40] Whitfield Diffie et Martin E. Hellman. « New Directions in Cryptography ». *IEEE Transactions on Information Theory*, vol. IT-22, nov. 1976. http://citeseer.ist.psu.edu/340126.html.

[41] Edsger Wybe Dijkstra. « The structure of the THE multiprogramming system ». *Communications of the ACM (CACM)*, vol. 11 nº 5, mai 1968. http://www.acm.org/classics/mar96/.

[42] Direction centrale de la sécurité des systèmes d'information. « Expression des Besoins et Identification des Objectifs de Sécurité », 2003. http://www.ssi.gouv.fr/fr/confiance/ebiospresentation.html.

[43] Gilles Dubertret. *Initiation à la cryptographie*. Vuibert, Paris, 2002.

[44] Albert Ducrocq et André Warusfel. *Les mathématiques – Plaisir et nécessité*. Vuibert, Paris, 2000. Plaidoyer pour une discipline malmenée, au moyen de nombreux exemples historiques et modernes auxquels l'érudition des auteurs et leur talent de vulgarisateurs confèrent un rythme trépidant et passionnant.

[45] Jean-Pierre Dupuy. *Pour un catastrophisme éclairé – Quand l'impossible est certain*. Éditions du Seuil, Paris, 2002.

[46] Kjeld Borch Egevang et Paul Francis. « RFC 1631 – The IP Network Address Translator (NAT) », mai 1994. http://www.ietf.org/rfc/rfc1631.txt.

[47] Carl Ellison et Bruce Schneier. « Ten Risks of PKI : What You're Not Being Told About Public Key Infrastructure ». *Computer Security Journal*, vol. 16, nº 1, 2000. http://www.schneier.com/paper-pki.html.

[48] David Evans. « What Biology Can (and Can't) Teach Us About Security ». *USENIX Security Symposium*, 12 août 2004. http://www.cs.virginia.edu/~evans/usenix04/usenix.pdf.

[49] Edward W. Felten. « DRM and Public Policy ». *Communications of the ACM (CACM)*, (vol. 48, nº 7), juillet 2005. http://www.csl.sri.com/users/neumann/insiderisks05.html#181.

[50] Richard P. Feynman. « Personal observations on the reliability of the Shuttle ». 1986. http://science.ksc.nasa.gov/shuttle/missions/51-l/docs/rogers-commission/Appendix-F.txt.

[51] Éric Filiol. *Les virus informatiques : théorie, pratique et applications*. Collection IRIS. Springer Verlag, Paris, 2003.

[52] Éric Filiol. « Le danger des virus blindés ». *La lettre – Techniques de l'ingénieur – Sécurité des systèmes d'information*, (6), novembre-décembre 2005.

[53] Éric Filiol. « Évaluation des logiciels antiviraux : quand le marketing s'oppose à la technique ». *MISC*, (21), octobre 2005. Dans un excellent numéro consacré aux *Limites de la sécurité*.

[54] Nicolas Fischbach. « Sécurité de la VoIP chez un opérateur ». 2006. http://www.

ossir.org/jssi/jssi2006/supports/1B.pdf.

[55] Gustave Flaubert. *Bouvard et Pécuchet*. Le Seuil, Paris, 1857. Comme il s'agit, en fin de compte, d'un livre sur la bêtise, sa lecture sera utile à quiconque se préoccupe de sécurité, puisque souvent les failles de sécurité ne sont pas sans lien avec la bêtise.

[56] Laurence Freyt-Caffin. « L'administrateur réseau, un voltigeur sans filet ». Dans *5e journées réseau JRES*, 2003. En ligne ici : http://2003.jres.org/actes/paper.130.pdf.

[57] Simson Garfinkel. *PGP* – Pretty Good Privacy. O'Reilly, Sebastopol, Californie (Paris), 1995. Traduction de Nat Makarévitch.

[58] Simson L. Garfinkel. « VoIP and Skype Security ». *Tactical Technology Collective*, 12 mars 2005. http://www.tacticaltech.org/skype_security.

[59] Solveig Godeluck. *La géopolitique d'Internet*. La Découverte, Paris, 2002. 247 pages.

[60] Katie Hafner et Matthew Lyon. *Where Wizards Stay Up Late – The Origins of the Internet*. Pocket Books, Londres, 1996.

[61] John L. Hennessy et David A. Patterson. *Computer Architecture : a Quantitative Approach*. Morgan Kaufman Publishers (Vuibert pour la traduction française), San Mateo, Calif., USA, 1996-2001. Ce livre donne à la description de l'architecture des ordinateurs une ampleur intellectuelle que peu soupçonnaient. En annexe, une bonne introduction à la représentation des nombres (norme IEEE 754 notamment). La traduction française est recommandable.

[62] Andrew Hodges. *Alan Turing : the Enigma (Alan Turing : l'Énigme de l'intelligence)*. Simon and Schuster (Payot, Paris pour la traduction), New-York, USA, 1983.

[63] Michael Howard et David LeBlanc. *Écrire du code sécurisé*. Microsoft, Redmond, USA, 2003. Traduction de Marc Israël.

[64] G. Dan Hutcheson. « The World Has Changed ». *VLSI Research*, 13 avril 2005. https://www.vlsiresearch.com/public/600203v1.0.pdf.

[65] ISO/IEC. « Information technology – Systems Security Engineering – Capability Maturity Model (SSE-CMM®) ». Norme internationale n° 21827, 2002.

[66] ISO/IEC. « Lignes directrices pour l'audit des systèmes de management de la qualité et/ou de management environnemental ». Norme internationale n° 19011, 2002.

[67] ISO/IEC. « Common Criteria for Information Technology Security Evaluation ». Norme internationale n° 15408, 2005.

[68] ISO/IEC. « Information Security Management Systems – Requirements ». Norme internationale n° 27001, 2005.

[69] ISO/IEC. « Information technology. Code of practice for information security management ». Norme internationale n° 17799, 2005.

[70] Kevin Krewell. « A Look Ahead To 2006 ». *Microprocessor Report*, vol. 20 n° 1,

janvier 2006. La revue mensuelle avec édition hebdomadaire sur le Web : http://www.mpronline.com/mpr/index.html du microprocesseur et de ses évolutions techniques et industrielles. Informée, compétente, beaucoup de détail technique exposé avec clarté.

[71] Benjamin A. Kuperman, Carla E. Brodley, Hilmi Ozdoganoglu, T.N. Vijaykumar, et Ankit Jalote. « Detection and Prevention of Stack Buffer Overflow Attacks ». *Communications of the ACM (CACM)*, vol. 48 n° 11, novembre 2005.

[72] Sophie Le Pallec. « La convergence des identifiants numériques ». Dans *Actes du congrès JRES*, 2005. http://www.jres.org/paper/70.pdf.

[73] Legalis.net. « Legalis.net ». 2 août 2006. http://www.legalis.net.

[74] Lawrence Lessig. *The future of ideas – The fate of the commons in a connected world*. Random House, New York, 2001. 352 pages.

[75] Steven Levy. *Hackers : Heroes of the Computer Revolution*. Doubleday, USA, 1984.

[76] Cédric Llorens, Laurent Levier, et Denis Valois. *Tableaux de bord de la sécurité réseau*. Eyrolles, Paris, 2006.

[77] Robert Longeon et Jean-Luc Archimbaud. *Guide de la sécurité des systèmes d'information – à l'usage des directeurs*. Centre National de la Recherche Scientifique (CNRS), Paris, 1999.

[78] Michael W. Lucas. *PGP & GPG - Assurer la confidentialité de ses e-mails et de ses fichiers*. Eyrolles (traduit par Daniel Garance), Paris, 2006.

[79] Alfred J. Menezes, Paul C. van Oorschot, et Scott A. Vanstone. *Handbook of Applied Cryptography*. CRC Press, Boca Raton, Floride, États-Unis, 1997. Une introduction complète au sujet, disponible en consultation sur le Web : http://www.cacr.math.uwaterloo.ca/hac/.

[80] Multicians. « Multics », 10 septembre 2006. http://www.multicians.org/.

[81] Stéphane Natkin. *Les protocoles de Sécurité d'Internet*. Dunod, Paris, 2002.

[82] Stephen Northcutt et Judy Novak. *Détection d'intrusion de réseau*. Vuibert, Paris, 2002 (2004 pour la traduction). Traduction de Raymond Debonne.

[83] Michael J. O'Donnell. « Separate Handles from Names on the Internet ». *Communications of the ACM*, vol. 48, n° 12, pp. 79-83, décembre 2005.

[84] Loïc Pasquiet. « Déploiement d'une solution de téléphonie sur IP dans un campus ». 2006. http://www.ossir.org/jssi/jssi2006/supports/2A.pdf.

[85] Jacky Pierson et Robert Longeon. « La biométrie (suite) ». *Sécurité Informatique*, avril 2004. Suite de l'article du bulletin de sécurité informatique du CNRS qui expose clairement les limites de la biométrie : http://www.sg.cnrs.fr/FSD/securite-systemes/revues-pdf/num48.pdf.

[86] W. Curtis Preston. *SANs and NAS*. O'Reilly, Sebastopol, California, 2002.

[87] Christian Queinnec. « le Filtrage : une application de (et pour) Lisp ». 1995. http://www-spi.lip6.fr/~queinnec/Books/LeFiltrage.ps.gz.

[88] Marcus J. Ranum. « The Six Dumbest Ideas in Computer Security ». *http://www.certifiedsecuritypro.com/*, 1er septembre 2005. http://www.ranum.com/security/computer_security/editorials/dumb/.

[89] Marcus J. Ranum. « What is *Deep Inspection ?* ». *Site de Marcus J. Ranum*, 6 mai 2005. http://www.ranum.com/security/computer_security/editorials/deepinspect/.

[90] Yakov Rekhter, Robert G. Moskowitz, Daniel Karrenberg, Geert Jan de Groot, et Eliot Lear. « RFC 1918 – Address Allocation for Private Internets) », février 1996. Cette RFC remplace les 1597 et 1627 de 1994 ; http://www.ietf.org/rfc/rfc1918.txt.

[91] Ronald Rivest, Adi Shamir, et Leonard Adleman. « A Method for Obtaining Digital Signatures and Public-Key Cryptosystems ». *CACM*, 21(2), février 1978. L'article fondateur, accessible en ligne ici : http://theory.lcs.mit.edu/~rivest/rsapaper.pdf.

[92] Mark E. Russinovich. *Windows Internals : Windows 2000, Windows XP & Windows Server 2003*. Microsoft Press, Redmond, État de Washington, 2005.

[93] Mark E. Russinovich. « Sony, Rootkits and Digital Rights Management Gone Too Far ». *Sysinternals*, octobre 2005. http://www.sysinternals.com/blog/2005/10/sony-rootkits-and-digital-rights.html.

[94] Alfred Rényi. *Calcul des probabilités*. Jacques Gabay [pour la traduction], Budapest [Paris], 1966. Ce livre qui a fait date dans son domaine contient un exposé particulièrement incisif et élégant de l'algèbre de Boole.

[95] Emmanuel Saint-James. *La Programmation applicative (de Lisp à la machine en passant par le λ-calcul)*. Hermès, Paris, 1993. Avec une préface de Jacques Arsac. Une étude riche et originale avec des aperçus saisissants sur la programmation.

[96] Olivier Salaün. « Introduction aux architectures Web de *Single-Sign On* ». 2003. http://2003.jres.org/actes/paper.116.pdf.

[97] Cliff Saran. « BP turns its back on traditional IT security with Internet access to company systems ». *Computer Weekly*, 3 septembre 2004.

[98] Hervé Schauer. « Site d'Hervé Schauer Consultants ». *Hervé Schauer Consultants*, 16 octobre 2005. http://www.hsc.fr/index.html.fr.

[99] Hervé Schauer. « VoIP et sécurité – Retour d'expérience d'audits de sécurité ». 2006. http://www.hsc.fr/ressources/presentations/tenor06-voip-sec/.

[100] Bruce Schneier. *Secrets et mensonges – Sécurité numérique dans un monde en réseau*. John Wiley & Sons (Vuibert pour la traduction française), New York (Paris), 2000 (2001). Traduction de Gabriel Otman et Jean-Jacques Quisquater.

[101] Security Focus. « Site de Security Focus ». *Security Focus*, 16 octobre 2005. http://www.securityfocus.org/.

[102] Avi Silberschatz, Peter Galvin, et Greg Gagne. *Principes appliqués des systèmes d'exploitation*. Vuibert (pour la traduction française), Paris, 2001.

[103] Simon Singh. *The Code Book (Histoire des codes secrets)*. J.-C. Lattès (pour la traduction française), Paris, 1999. Un ouvrage de vulgarisation passionnant.

[104] Sophos. « Rapport Sophos 2005 sur la gestion des menaces à la sécurité », 2005.

[105] Pyda Srisuresh et Kjeld Borch Egevang. « RFC 3022 – Traditional IP Network Address Translator (Traditional NAT) », janvier 2001. `http://www.ietf.org/rfc/rfc3022.txt`.

[106] Richard M. Stallman. « Pouvez-vous faire confiance à votre ordinateur ? ». *Logiciel libre, société libre : articles choisis de Richard M. Stallman*, 2002. `http://www.gnu.org/philosophy/can-you-trust.fr.html`.

[107] Michael Szydlo. « SHA-1 Collisions can be Found in 2^{63} Operations ». *RSA Laboratories*, 19 août 2005. `http://www.rsasecurity.com/rsalabs/node.asp?id=2927`.

[108] Andrew S. Tanenbaum. *Réseaux*. Pearson Education (pour la traduction française), Paris, 2003.

[109] Robert Bruce Thompson et Barbara Fritchman Thompson. *PC Hardware in a Nutshell*. O'Reilly, Sebastopol, Calif., USA, 2003. Un ouvrage pratique indispensable. Vous comprendrez rétrospectivement la cause de tous vos ennuis avec la gravure de CD-Roms, la géométrie des disques durs... ou la sécurité.

[110] Isabelle N. Tisserand. *Hacking à cœur – Les enfants du numérique*. Éditions e/dite, Paris, 2002.

[111] Roland Topor. *Le sacré livre de Prouto*. Syros, Paris, 1990.

[112] Vernor Vinge. *True Names*. Tor Books, USA, 1981.

[113] Michel Volle. « Histoire d'un tableau de bord ». 20 novembre 2002. `http://www.volle.com/travaux/tdb.htm`.

[114] Michel Volle. *e-conomie*. Economica, Paris, 2000. Une analyse économique informée et pénétrante des nouvelles technologies par un maître de l'économétrie et de la statistique, disponible en ligne ici : `http://www.volle.com/ouvrages/e-conomie/table.htm`.

[115] Michel Volle. *De l'informatique*. Economica, Paris, 2006.

[116] Michel Volle. « Histoire d'un *datawarehouse* ». 21 mars 2003. `http://www.volle.com/travaux/dwh.htm`.

[117] Xiaoyun Wang, Andrew Yao, et Frances Yao. « New Collision search for SHA-1 ». Dans *Crypto'05*, 2005.

[118] Xiaoyun Wang, Yiqun Lisa Yin, et Hongbo Yu. « Finding Collisions in the Full SHA-1 ». Dans *Advances in Cryptology – Crypto'05*, 2005. `http://www.infosec.sdu.edu.cn/paper/sha1-crypto-auth-new-2-yao.pdf`.

[119] Gerald M. Weinberg. *The Psychology of Computer Programming*. Van Nostrand

Reinhold, New York, 1971.
[120] Wikipédia. « Débordement de tampon ». *Wikipédia*, 14 octobre 2005. http://fr.wikipedia.org/wiki/Buffer_overflow.
[121] Wikipédia. « Poste à poste ». *Wikipédia*, 15 novembre 2005. http://fr.wikipedia.org/wiki/Poste-à-poste.
[122] Wikipédia. « SHA-1 ». *Wikipédia*, 15 novembre 2005. http://fr.wikipedia.org/wiki/SHA-1.
[123] Wikipédia. « Network address translation ». *Wikipédia*, 19 décembre 2005. http://fr.wikipedia.org/wiki/NAT.
[124] Philippe Wolf. « De l'authentification biométrique ». *Sécurité Informatique*, octobre 2003. Cet article du bulletin de sécurité informatique du CNRS expose clairement les limites de la biométrie : http://www.sg.cnrs.fr/FSD/securite-systemes/revues-pdf/num46.pdf.

Index

A

B

C

Q

R

S

Sécurité informatique



L. Bloch

Ancien élève de l'École nationale de la statistique et de l'administration économique (ENSAE), Laurent Bloch a travaillé à l'INSEE et dirigé les services d'informatique scientifique de l'INED, du CNAM et de l'Institut Pasteur. Aujourd'hui responsable de la sécurité des systèmes d'information de l'INSERM, il est *Lead Auditor* certifié IS 27001. Il est l'auteur des livres *Systèmes d'information, obstacles et succès* et *Les systèmes d'exploitation des ordinateurs* (Vuibert, 2005 et 2003).

C. Wolfhugel

Ingénieur INSA de Lyon, Christophe Wolfhugel s'est spécialisé dès la fin des années 1980 dans les réseaux IP, notamment l'Internet, et les services associés. Il est aujourd'hui expert dans ces domaines dans la division Orange Business Services de France Télécom.

Comprendre les menaces informatiques pour les juguler

L'administrateur et le responsable informatique affrontent une insécurité informatique protéiforme et envahissante, qui menace tant les données que les applications de l'entreprise : virus, attaques par le réseau, tromperie sur le Web, etc. Bien des outils sont proposés pour y faire face, mais encore faut-il comprendre leur rôle et leur mode opératoire et les replacer dans le cadre d'une politique de sécurité efficace. On devra pour cela garder en tête les principes qui animent tout système d'information et chasser de dangereuses idées reçues.

Une approche systématique de la sécurité informatique

Écrit par le responsable de la sécurité des systèmes d'information de l'INSERM, ce livre limpide expose les causes des risques inhérents à tout système informatique – et les moyens de s'en protéger. S'adressant aux administrateurs et responsables de systèmes d'informations comme à leurs interlocuteurs, il offre au professionnel consciencieux des principes clairs et une méthode rigoureuse pour concevoir une véritable politique de sécurité.

Au sommaire

À qui s'adresse cet ouvrage ?

– Aux administrateurs de systèmes et de réseaux, mais aussi aux DSI et aux responsables de projets
– À tous ceux qui doivent concevoir ou simplement comprendre une politique de sécurité informatique

Code éditeur : G12021
ISBN : 2-212-12021-4
ISBN 13 : 978-2-212-12021-9

Conception : Nord Compo